

PATENT ABSTRACTS OF JAPAN

(11)Publication number : 05-334072

(43)Date of publication of application : 17.12.1993

.....
(51)Int.Cl. G06F 9/06

.....
(21)Application number : 03-308350 (71)Applicant : INTERNATL BUSINESS
MACH CORP <IBM>

(22)Date of filing : 29.10.1991 (72)Inventor : ROBERT CARL BEECHER
MICHAEL JOSEPH CORRIGAN
FRANCIS JOSEPH RIADON JR
JAMES WILLIAM MORAN

.....
(30)Priority

Priority number : 90 629295

Priority date : 14.12.1990

Priority country : US

.....
(54) DEVICE AND METHOD FOR MANAGING USE OF SOFTWARE

(57)Abstract:

PURPOSE: To provide an improved method and device for managing the use of software
in a computer system.

CONSTITUTION: Software is distributed with no execution qualification and can only be executed with a separately distributed enciphered qualified key 111. The key 111 contains the serial number 410 of a computer to which the license of the software is given and a plurality of qualifying bits indicating the software module qualified to be run on a machine. The distributed software contains a plurality of qualification verifying triggers. Each trigger is an object code type of single machine word instruction which discriminates the product number of a software module. Therefore, an improved method and device for managing the use of software in a computer system can be provided.

LEGAL STATUS [Date of request for examination] 29.10.1991

[Date of sending the examiner's decision of rejection]

[Kind of final disposal of application other than the examiner's decision of rejection or application converted registration]

[Date of final disposal for application]

[Patent number] 2066371

[Date of registration] 24.06.1996

[Number of appeal against examiner's decision of rejection]

[Date of requesting appeal against examiner's decision of rejection]

[Date of extinction of right] 25.10.1999

* NOTICES *

JPO and NCIPi are not responsible for any damages caused by the use of this translation.

1.This document has been translated by computer. So the translation may not reflect the original precisely.

2.**** shows the word which can not be translated.

3.In the drawings, any words are not translated.

CLAIMS

[Claim(s)]

[Claim 1] A means to give rating for performing a software module to computer system, The trigger means which has been arranged in the above-mentioned software module and with which the plurality which carries out the trigger of the verification of rating became independent, A rating verification means to verify that answer each of the trigger means with which the above-mentioned plurality became independent, and the above-mentioned computer system has rating for performing the above-mentioned software module, When the above-mentioned rating verification means is answered, the above-mentioned computer system did not have rating for performing the above-mentioned software module and the above-mentioned rating verification means judges Equipment for managing use of the software module performed with computer system equipped with a means to close activation of the above-mentioned software module.

[Claim 2] Equipment for managing use of a software module according to claim 1 whose trigger means with which the above-mentioned plurality became independent is the single object code instruction which carries out the trigger of

the above-mentioned rating verification means.

[Claim 3] It is equipment for managing use of the software module according to claim 2 with which the above-mentioned additional step is not performed, but the above-mentioned software module is no longer performed appropriately, when the above-mentioned software module is corrected by also performing the additional step which needs for suitable activation of the above-mentioned software module the above-mentioned object code instruction which carries out the trigger of the above-mentioned rating verification means, consequently removing the above-mentioned object code instruction.

[Claim 4] Equipment for managing the use of a software module according to claim 2 whose above-mentioned object code instruction which carries out the trigger of the above-mentioned rating verification means contains the part number of the above-mentioned software module.

[Claim 5] The equipment for managing use of a software module according to claim 4 contain a means access to the above-mentioned rating grant information included in the entry in the product lock table in the above-mentioned computer system on which the above-mentioned rating verification means is equipped with two or more entries with which each is related to a part number, including rating

grant information, and the above-mentioned product lock table relevant to the above-mentioned part number which answers the above-mentioned object code instruction and is contained in the above-mentioned object code instruction.

[Claim 6] Equipment equipped with a means by which an above-mentioned means to give rating for performing the above-mentioned software module to the above-mentioned computer system generates the rating grant key which consists of data, and a means to input the above-mentioned rating grant key into the above-mentioned computer system for managing use of a software module according to claim 1.

[Claim 7] Equipment for managing use of a software module according to claim 6 which an above-mentioned means by which the above-mentioned computer system generates a rating grant key including an original identifier generates the above-mentioned rating grant key using the above-mentioned original identifier, and gives rating for performing software only on the computer system with which the above-mentioned rating grant key contains the same original identifier.

[Claim 8] The means prevent from decoding only on the computer system in which the enciphered rating grant key which enciphers the above-mentioned rating grant key using the above-mentioned original identifier, and is obtained as

a result has the same original identifier, A means in computer system to access the original identifier of the above-mentioned computer system, Equipment equipped with a means in the above-mentioned computer system to answer an above-mentioned means to access the above-mentioned original identifier, and to decode the rating grant key as which the above was enciphered for managing use of a software module according to claim 7.

[Claim 9] The phase which arranges the trigger means with which the plurality which carries out the trigger of the verification of rating became independent in a software module, When it meets with one of the trigger means with which plurality became [above-mentioned] independent during the phase of performing the above-mentioned software module, and activation of the above-mentioned software module, By the phase which carries out the trigger of the rating verification actuation which verifies that computer system has rating for performing the software module in the above-mentioned computer system, and the above-mentioned rating verification When judged with there being no rating for performing the above-mentioned software module in the above-mentioned computer system, The approach for managing the use including the phase of closing activation of the above-mentioned software

module of the software module performed on computer system.

[Claim 10] The approach including arranging the single object code instruction whose above-mentioned phase which arranges the trigger means with which plurality became independent in the above-mentioned software module carries out the trigger of the above-mentioned rating verification actuation to two or more separate locations in the above-mentioned software module for managing use of a software module according to claim 9.

[Claim 11] It is an approach for managing use of the software module according to claim 10 with which the above-mentioned additional step is not performed, but the above-mentioned software module is no longer performed appropriately, when the above-mentioned software module is corrected by also performing the additional step which needs for suitable activation of the above-mentioned software module the above-mentioned object code instruction which carries out the trigger of the above-mentioned rating verification actuation, consequently removing an object code instruction.

[Claim 12] The approach for managing use of a software module according to claim 10 the above-mentioned object code instruction which carries out the trigger of the above-mentioned rating verification actuation contains the part

number of the above-mentioned software module.

[Claim 13] The approach including the above-mentioned phase which carries out the trigger of the rating verification actuation accessing the above-mentioned rating grant information included in the entry in the product lock table relevant to the part number contained during the above-mentioned object code instruction including the phase maintain the product lock table equipped with two or more entries with which each is related to a part number, including rating grant information within the above-mentioned computer system for managing use of a software module according to claim 12.

[Claim 14] The approach for managing use of a software module according to claim 9 which consists of two or more data bits, and includes the phase which generates the rating grant key which offers the information which enables it to judge whether the above-mentioned computer system has rating for performing the above-mentioned software module, and the phase of inputting the above-mentioned rating grant key into the above-mentioned computer system.

[Claim 15] The method for managing use of a software module according to claim 14 of the above-mentioned phase where the above-mentioned computer system generates a rating grant key including an original identifier generating

the above-mentioned rating grant key using the above-mentioned original identifier, and giving rating for performing software only on the computer system with which the above-mentioned rating grant key contains the same original identifier.

[Claim 16] Have a means to receive rating for performing a software module, and the trigger means in the above-mentioned software module is answered. On the computer system which has a rating verification means to verify that the above-mentioned computer system has rating for performing the above-mentioned software module it is the program product for managing grant of rating with which the above-mentioned software module is performed, and was recorded on the record medium -- with the software module of a piece at least A program product equipped with the trigger means which carries out the trigger of the verification of rating on computer system and with which the plurality in the above-mentioned software module became independent.

[Claim 17] The program product for managing rating grant according to claim 16 which is the single object code instruction of the trigger means with which plurality became [above-mentioned] independent which carries out the trigger of the verification of the above-mentioned rating, respectively.

[Claim 18] It is a program product for managing the rating grant according to claim 17 by which the above-mentioned additional step is not performed but a software module is no longer performed appropriately, when the above-mentioned software module is corrected by also performing the additional step which needs for suitable activation of the above-mentioned software module the above-mentioned object code instruction which carries out the trigger of the above-mentioned rating verification means, consequently removing the above-mentioned object code instruction.

[Claim 19] A program product for the above-mentioned object code instruction which carries out the trigger of the above-mentioned rating verification means to manage [the above-mentioned rating verification means on the above-mentioned computer system / each] the rating grant according to claim 17 which has the product lock table equipped with two or more entries in relation to a part number in computer system including the part number of the above-mentioned software module, including rating grant information.

[Claim 20] Have a means to receive rating for performing a software module, and the trigger means in the above-mentioned software module is answered. On the computer system which has a rating verification means to verify that the

above-mentioned computer system has rating for performing the above-mentioned software module The phase which arranges the trigger means with which it is the method of distributing a software module that the above-mentioned software module is performed, and the plurality which carries out the trigger of the verification of rating became independent in the above-mentioned software module, The method including the phase of distributing the above-mentioned software module to the above-mentioned computer system, and the phase which gives rating for performing the above-mentioned software module to the above-mentioned computer system of distributing a software module.

[Claim 21] The method including the above-mentioned phase which arranges the trigger means with which plurality became [above-mentioned] independent in the above-mentioned software module arranging the single object code instruction which carries out the trigger of the verification of the above-mentioned rating in two or more separate locations in the above-mentioned software module of distributing a software module according to claim 20.

[Claim 22] It is the approach of distributing the software module according to

claim 21 with which the above-mentioned additional step is not carried out, but a software module is no longer appropriately performed when the above-mentioned software module is corrected by also performing the additional step which needs for suitable activation of the above-mentioned software module the above-mentioned object code instruction which carries out the trigger of the above-mentioned rating verification, consequently removing the above-mentioned object code instruction.

[Claim 23] The approach the above-mentioned object code instruction which carries out the trigger of the above-mentioned rating verification distributes [the above-mentioned rating verification means on computer system / each] the software module according to claim 21 which has the product lock table equipped with two or more entries in relation to a part number in computer system including the part number of the above-mentioned software module, including rating grant information.

[Claim 24] The approach including the phase which generates a rating grant key the above-mentioned phase which gives rating for performing the above-mentioned software module to the above-mentioned computer system consists of two or more data bits, and offer the information which enables it to

judge whether the above-mentioned computer system has rating for performing the above-mentioned software module, and the phase transmit the above-mentioned rating grant key to computer system of distributing a software module according to claim 20.

[Claim 25] The method of distributing a software module according to claim 24 of the above-mentioned phase where the above-mentioned computer system generates a rating grant key including an original identifier generating the above-mentioned rating grant key using the above-mentioned original identifier, and giving rating for performing software only on the computer system with which the above-mentioned rating grant key contains the same original identifier.

[Claim 26] The method of distributing a software module according to claim 20 that the above-mentioned phase which gives rating for performing the above-mentioned software module to the above-mentioned computer system including distributing two or more software modules which have the trigger means with which the plurality to which the above-mentioned phase distribute the above-mentioned software module carries out the trigger of the rating verification on a respectively single storage became independent includes giving

separate rating to at least two of two or more above-mentioned software

modules on the above-mentioned single storage.

DETAILED DESCRIPTION

[Detailed Description of the Invention]

[0001]

[Industrial Application] This invention relates to a computer user more specifically restricting that license software can be used in the form where a license is not followed about use of computer software.

[0002]

[Description of the Prior Art] Present-day computer system is the very complicated machine which took in the result of skill on the technique covering the time amount which it cannot finish counting, and programming to the design. They can be divided roughly into hardware and software although computer system contains many components. Hardware is the material circuit and board which constitute a system, a cable, storage, enclosure, etc. However, the same with the ability of the scenario which can win the Pulitzer prize with a typewriter

not to be written, hardware is itself and cannot solve the problem in the actual world. Hardware needs the instruction which tells what should be carried out. "Software" means the instruction which makes hardware perform useful work in the reasonable pure form. (However, it may be vaguely applied to the medium by which software is memorized and distributed) . Software as well as hardware is the product of human being's originality and creativity. The software of high quality needs the remarkable creativity by the side of the implementer called a programmer, training, and intelligence. Many universities have prepared the curriculum of "computer science" and a similar subject of study in order to teach people the technique which creates software. This whole industry held thousands of persons' carrier, and by the time it creates the software which does useful work, it will have grown up. As a result of spending an immense effort on development of software, the worth of software which is some computer system of exceeding worth of hardware is not new, either.

[0003] One of the descriptions of present-day computer system is being able to transmit and copy data easily at high speed very much. Generally, this is the need and useful capacity. However, this means that the created software to which the effort was applied can reproduce with what [for 1 second / 1/] with a

comparatively cheap magnetic medium, and it may use improperly for many years again. People who have not got authorization can reproduce worthy software by few costs and efforts. Generally this custom is called software property infringement. In order to control software property infringement in recent years, various laws which impose the duty on Penal Code and Civil Code have been enacted. However, since it is cut by temptation [software can copy comparatively easily, and] to come to do so since software is expensive, the literary piracy of software still poses a problem.

[0004] In the case of the cheap software with which extensive distribution is carried out for [which is called a "personal computer"] small computers, usually, the license to use of software is accepted at the lump sum payment tariff which the same amount fixed to all customers. This is not performed so much in a large-sized mainframe computer system. Such software for large-sized systems may require the code of millions of lines, and needs very a lot of investment for development and maintenance. If fixed sufficient single lump sum payment tariff for a developer to collect these investment is set up, it tends to become so expensive for many petty users that it cannot be used. Therefore, in the case of a mainframe, usually, the charge of a license of software based on the amount

used is charged. Such one approach called "a gradual price setup" charges the tariff based on the engine performance of a customer's machine according to the adjustable tariff structure. The charge of a license higher than the customer of the machine with which the engine performance is as inferior as the customer who more terminals are connected and uses the machine of a reliance high speed to the same software is paid. Another usual custom charges a tariff separately to software maintenance upgrade.

[0005] Considering the level of a know how required to create high-definition software, and the amount of the time amount concerning it, and an effort, money is required for creation of such software. When the developer of software is not rewarded as compared with the training and efforts, those who create software stop there being. It is not only actually the upper request to protect the investment to the software which the developer performed, but it is also a moral request. Therefore, it is difficult for the just owner of software to use it without authorization of software, and he has asked for development of the software distribution approach that the developer of software is justly rewarded to the product.

[0006] Software is distributed by the approach a large number differ from a just

owner. These distribution approaches can be divided roughly into three groups, the unrestricted rating giving method, the limit rating giving method, and the non-rating giving method, from a viewpoint which prevents unauthorized use.

[0007] Unrestricted rating grant means that the distributed software runs without a limit by every machine which was the object of the design. The owner who distributes the software of unrestricted rating grant has to distribute only the software which has rating for a user paying a countervalue to it and performing it to each user. It is only legal and the contractual obligation which bar reproducing in the form where this is not permitted to the receiving area of such software, or using it. By the cheap software to which it is licensed at a lump sum payment tariff, it is the distribution approach which is used with most personal computers that this is the most ordinary.

[0008] Limit rating grant means that software builds in a certain kind of limit of restricting a user copying it and being able to perform on without limit many machines. There is the limit rating giving method some differ. One of them is a copy limit which restricts the number of the copies which can be created from the distributed original. Although a copy limit realizes protection of a certain level to software property infringement, it has some fault. It is necessary to distribute

only the software which has rating for the user also performing [an owner] a copy limit to each user like unrestricted rating grant. This strikes not a full proof but a protection feature, and the program which can take the literal copy of the software with which copy protection of the bush was carried out also exists. Finally, this prevents him from the ability for a user to take a just copy or run software from high-speed memory for the purpose of backup. Another limit rating giving method codes information peculiar to a user or a machine in the software itself. In case software is performed, a machine inspects in order to confirm that software is permitted to the machine or user. This approach realizes protection, without barring that a user can take a just copy. However, this compiles each copy of the distributed software uniquely, respectively, and puts it on a distribution medium, and since it must ship, a very expensive distribution system is needed.

[0009] The non-rating giving method means that the distributed software needs the rating grant separately distributed for performing by being made the disable. A certain non-rating giving method may avoid distribution of custom software completely. However, such not all approaches necessarily have such capacity. For example, an owner can distribute the software program of a majority of same

set to all those customers on a single generic name medium, and can distribute separately the authorization key which allows performing only the program whose customer finished payment and which turned the individual exception to each customer. Although the non-rating giving method avoids many problems which accompany other approaches, a current design has a large possibility of receiving forgery and the remarkable performance degradation of rating grant. When the most, the device in which it is used in order to prohibit the rating grant for performing software needs to concentrate a rating verification means in a software module (as data or an instruction), in order to avoid the overhead on verification called performance degradation. Depending on the case, the overhead of this rating grant may be based on mounting of the size of a product identifier, and the protection routine in the distributed software. Or while the overhead makes it run software, it may be based on the need of performing a complicated decode means again. As a result of concentration of a such rating inspection, when an experienced programmer makes an invalid the element as which "patching, i.e., object code," was chosen, it is comparatively easy to make a protection feature into an invalid. When another, object code does not perform rating verification but the insurance call path of identifying a module by creating

a bit signature from it performs it. This causes the serious performance degradation of a call device unescapable, although patching is made hard to receive.

[0010] The protection approach currently taught with the conventional technique makes the compromise of protection level and facility. Although it is possible to obtain protection of a high level comparatively by coding information peculiar to a machine in software, each software copy distributed must pay the sacrifice used as an original thing of maintaining a very complicated distribution system. Although cheaper distribution is also possible, the sacrifice that a part of protection is lost must be paid. Possibility that will realize protection of a high level, and can distribute easily using extensive distribution technique, and the engine performance of a system and a user will take the copy for a backup copy justly, and the approach which does not bar other required functions unfairly are searched for. Coincidence is asked also for a gradual price setup and the approach of supporting the separate charge of a license for every version that different software differs.

[0011]

[Problem(s) to be Solved by the Invention] The purpose of this invention is to

offer the approach and equipment which manage use of the software in computer system and which have been improved.

[0012] Other purposes of this invention are in the rather than thing for which protection of a high level is offered to the unauthorized software use in computer system.

[0013] Other purposes of this invention are to reduce the costs which protect software to unauthorized use.

[0014] Other purposes of this invention are to reinforce the engine performance of computer system of performing software protected to unauthorized use.

[0015] Other purposes of this invention are to simplify the distribution system of the distribution person of software protected to unauthorized use.

[0016] Other purposes of this invention have such protection in offering the approach and equipment which reduce the effects affect just use of software and which protect software from unauthorized use.

[0017] Other purposes of this invention are to offer the approach and equipment which protect software from unauthorized use and which have been improved, while a user allows taking the just copy for backup of software.

[0018] Other purposes of this invention are to make it difficult to change software

so that unauthorized use may be attained.

[0019] Other purposes of this invention are to offer the approach and equipment which distribute the software which carried out a price setup gradually and which have been improved.

[0020]

[Means for Solving the Problem] According to this invention, software is distributed without the rating grant for performing. Activation of software is attained by the enciphered rating grant key which is distributed separately. This rating grant key contains two or more rating grant bits which instruct it to be the serial number of the machine with which software is licensed to it which software module to have rating it runs by that machine. The insurance decode device is built in the machine. An insurance decode device takes out the serial number of a machine, and this is used as a key for decoding a rating grant key. Subsequently, rating grant information is memorized by the product lock table in memory.

[0021] The distributed software contains two or more rating verification triggers. In the desirable example, each trigger is a simple-machine word instruction in object code which identifies the part number of a software module. If it meets

with such a rating verification trigger during activation of a software module, a machine will inspect the entry of the product lock table corresponding to the part number of a software module. When it has rating a product runs, activation is continued normally, and activation is closed when that is not right. Since this verification requires only one machine instruction, it can perform without hardly affecting the system-wide engine performance. Consequently, most number of such rating verification triggers can be arranged in object code, and when someone does "patching" of this trigger, it can make it impossible to change a code as a matter of fact. In the another example, a rating verification trigger also carries out work with or [useful in any way] which is the need to proper activation of a software module. By this, it becomes much more difficult to carry out "patching" of the software, and the effect to the engine performance of such a verification trigger is mitigated further.

[0022] Since the software itself does not include any rating grants, a limit of distribution is not required. In the desirable example, the distribution person of software can record two or more software modules on a single generic name medium, and can distribute 1 set of recorded same modules to all those customers. Each customer receives the original rating grant key to which he can

perform only the software module to which it is licensed. Since it cannot be performed without a suitable key even if a certain module with which it is not licensed to a customer is given, it does not become a big problem. A customer can load software to the store of others of his own system freely, or can create the copy for backup of software without limit.

[0023]

[Example] The explanatory view of the main elements of the software protection feature based on the desirable example of this invention is shown in drawing 1 .

A customer's computer system 101 is not [being / which was closely combined with control storage 103 / 102 / the arithmetic and program control (CPU), random access system being / 104 / the memory, and] eliminable, and contains electronically the original identifier 105 in which read is possible, and two or more storage 106, 107, and 108. In the desirable example, the original identifier 105 is the serial number of a machine. It is also possible to use other storage techniques in the desirable example, although stores 106-108 are rotating type magnetic disk memory. Computer system 101 also contains the console 109 and the software-media reader 110 which receive the input from an operator again. Although one set of a console and one software-media reader and three

storage are shown in drawing 1 , please understand that the actual number of such equipment attached in a system 101 is adjustable. Moreover, please understand that additional equipment is attached by the system 101. In the desirable example, although computer system 101 is AS/400 computer system of IBM, other computer system can be used.

[0024] A software module is distributed apart from a rating grant key (entitlement key) required to operate it. Originally, a software module is created on the computer system 125 for development. The computer system 125 for development contains the compiler 126 and the translator 127. A software module is recorded on the software record medium 112, is stored in a warehouse 120, and is distributed to a customer from there. The enciphered rating grant key 111 is distributed from a distribution person's selling office 121. The selling office 121 can access the computer system 124 for marketing. The computer system 124 for marketing includes generation/encryption program 122 of a rating grant key, and the data base 123 including customer information. Specifically, the data base 123 includes the information on the serial number of a machine required for generation of the enciphered rating grant key, and the class of processor. In the desirable example, the computer system 124 for

marketing is a computer which is located in the center and communicates with two or more selling offices. However, the computer system 124 for marketing is located in the selling office itself, and can also access the data base of a part or a center. Although two separate computer system 124 and 125 under a distribution person's control is shown in drawing 1 , please understand that that single computer system may be physically used performs both functions.

[0025] The enciphered rating grant key 111 is sent to a customer with mailing, a telephone, or other suitable means from the distribution person of software. Although it is possible to transmit a rating grant key on magnetic media, such as a floppy disk, electronically; a key is so sufficiently brief that an operator types on a console 109 and can input this into a system 101.

[0026] In the desirable example, many software modules are distributed on the same medium 112. A distribution person can give independently rating for using each module through a rating grant key. A warehouse 120 stores 1 set of software modules generically on a medium 112. 1 set of same generic software modules are shipped irrespective of whether the license which uses which module for each customer is given. The rating grant key distributed separately includes information for a system 101 to determine whether there is any rating

for performing which software module on it.

[0027] In the desirable example, software media 112 consist of one sheet or two or more read-only optical disks, and the medium reader 110 is an optical disk reader. However, please understand that an electronic formula distribution medium and other distribution media can also be used. If software media 112 are received, a customer will load a desired software module to a system 101 from the medium reader 110, and will usually memorize the software module to storage 106-108. A customer can generate one or more copies for backup of a software module on the suitable medium of arbitration, and can memorize these to an archive. A copy being freely possible and loading are possible for software media 112 excluding the limit to the copy and loading to a system.

[0028] The contents of the rating grant key 200 before encryption by the desirable example are shown in drawing 2 . This key contains the tariff group field 201, the software version field 202, the key type field 203, the machine serial number field 204, and the product rating grant flag 205. The tariff group field 201 is used for specifying one and supporting a gradual price setup of software from 16 kinds of machine phase values. The software version field 202 specifies the version level of the software with which rating is given. The thing for

which a tariff is imposed separately in order to maintain upgrade of software is expected. The version specified in the rating grant key 200 gives rating to the software of former (low order) total level from the version level. The key type field 203 is the field suspended for the number escape of a different product with which a key format and future modification of the relevance of a key are sake [modification] or supported. The machine serial number field 204 contains the serial number of the target machine [key / rating grant]. The product rating grant flag 205 is the 80-bit field where each contains 80 separate product flags corresponding to a part number. When rating is given to the corresponding part number, this bit is set to "1", and it is set to "0" when that is not right.

[0029] In the desirable example, a software module is distributed as compiled object code. The typical software module 300 is shown in drawing 3 . This software module includes two or more object code instructions which can be executed on computer system 101. According to this invention, some rating verification trigger instructions (the following, a "rating verification trigger", and notation) 301 are built into object code. All the rating verification triggers 301 contained in a software module are the same. The rating verification trigger 301 includes an operation code field 302, the version field 303, and the part number

field 304. The field 305 is not used. An operation code field 302 is the verb part of the object code instruction which identifies the actuation which should be performed. The version field 303 identifies the version level of a software module. The part number field 304 identifies the part number relevant to the software module. Although the information on a version and a part number changes for every module, single instruction code is used for all rating verification triggers. In the another example, a rating verification trigger is also the direct instruction (instruction without the need of dividing, and the operand for the processing being under instruction, and specifying) which performs other useful work of a certain. In such the another example, if a trigger instruction is executed, a system 101 will perform other actuation of a certain to rating verification and coincidence.

[0030] Computer system 101 includes a means to receive and decode the enciphered rating grant key 111, and a means to verify having rating for answering a rating verification trigger and a system performing a software module. In the desirable example, these functions are divided between the system hardware of different level, and the system software as shown in drawing 4 . In this example, the system 101 includes the hardware/software function of 4

level called the hardware level 401, the horizontal-microcode level 402, the executable code level 403, and the virtual-machine level 404. The machine interface 405 has separated virtual-machine level from all low-ranking level. The machine interface 405 is an interface which at least the bottom defined to the customer has in level. That is, although the order bit of virtual-machine level is defined to a customer, actuation of the level below it is not defined. Therefore, a customer does not have the capacity to change the instruction of the level below machine interface level directly. The machine serial number 410 unchangeable [eternal] is stored in the hardware level 401.

[0031] The horizontal microcode 402 contains the microcode entry which interprets the instruction set which can be performed. This is physically memorized by control storage 103, i.e., a desirable example, by the read-only memory (ROM) in which the alteration by the customer is impossible. The entry in a horizontal microcode is supporting the machine-key acquisition function 420, the lock setting up function 421, and the lock checking feature 422. The machine-key acquisition function 420 takes out the original identifier based on the system serial number from a certain eternal hardware location in the desirable example. The lock setting up function 421 accesses the product lock

table 460, and changes the entry in a table. A lock setting up function is the only microcode function in which the product lock table 460 can be changed. The lock checking feature 422 verifies whether the product lock table 460 is accessed, one of the entries is read, and rating is given.

[0032] The executable code level 403 includes the monitor support of a lower level, and the support which carries out the instruction defined by the machine interface. Although this is physically memorized by memory, since the internal specification is not defined to the customer, it is unchangeable in practice with a customer. The lock discharge routine 430, the initial-program-load (IPL) lock re-verification routine 431, and the exception-handling routine 432 are contained in the support of a lower level. The unlocking routine 430 decodes the rating grant key 111 using the machine key of a proper, and memorizes the enciphered rating grant key 111 on the product key table 450. If a system is initialized again, the initial-program-load lock re-verification routine 431 will take out the contents of the coded product key table 450, and will reconstruct the product lock table 460. The exception-handling routine 432 answers the exception condition generated by the function of the horizontal-microcode level 402. The executable code level 403 contains the software module 300 of an object code format.

[0033] Refer to the software for the virtual-machine level 404 as what is expressed by machine instruction. The computer of this level operates as a virtual machine in the semantics that immediate execution is not possible for machine instruction. Since at least the bottom which can be used for a customer is the interface of level, although the module of the object code format which can be performed on a system is created, the compile path of a non-conventional type is required for the virtual-machine level 404. This compile processing is explained later. Strictly, although the software created through the compile path of this non-conventional type must take the format of the object code which it is going to perform and which can be performed, it is common that the absolute instruction of virtual-machine level expresses and discusses, and it is more intelligible as if immediate execution was possible for the absolute instruction. If this is established, it can be said that a user's software which the virtual-machine level 404 compiled is included. This also includes [rather than] the support of the operating system of an upper level to the system 101 again. The support of this operation system contains two user interface routines required to support the input of a rating grant key on the virtual-machine level 404. The general input routine 441 is used for processing an input in normal operation. Furthermore, the

install input routine 440 special to inputting a rating grant key is required during initial installation of an operating system. The thing this [whose] is the need is because the part of the operating system of a high order is treated as other program products by this invention from the machine interface level 405. That is, such a part has a part number and the object code is influenced of a rating verification trigger. The install input routine 440 is the only part without the rating verification trigger of an operating system, therefore in case it introduces a system first, it can input a rating grant key.

[0034] Software modules 300 are some program products of the compiled object code format which is performed on a system 101. This means that it is accessible and exists in other objects of virtual-machine level as an entity of the virtual-machine level 404. However, the code which can actually be performed operates on the executable code level 403 as shown by the frame of a broken line. The executable code contains the rating verification trigger 301 (only one is shown in drawing) performed by the lock checking feature 422 of a horizontal microcode.

[0035] The coded product key table 450 is shown in drawing 5 . Recovering is possible, when it must intercept the power of a system or reinitialization must be

carried out by other reasons, since the product key table 450 is contained in random access memory 104 and is reproduced by the nonvolatile storage. The table 450 contains 80 entries 501 to which each can respond to each part number. Each entry 501 includes the perfect copy of the enciphered rating grant key 502 which is applied to the part number of the entry, the time stamp 503 in which it is shown when the key was used first, the version number 504 and the tariff group 505 of a key of the key, and the rating grant bit 506 which shows whether the key unlocks a product. The time stamp 503 can be enciphered. A version number 504, the tariff group 505, and the rating grant bit 506 repeat the information included in the enciphered rating grant key 502. They are contained in this table in order to support collating. However, unless it is after verifying the information in the rating grant key 502 enciphered first, the entry of the product lock table 460 cannot be changed and rating of program execution cannot be given. Since each rating grant key 502 in the product key table 450 is the enciphered form, in order to prevent access to this table of a user, it does not need a special measure.

[0036] The product lock table 460 is shown in drawing 6 . This table is contained in the special lower address range of random access memory 104 under perfect

control of a horizontal microcode. The product lock table 460 contains 80 entries 601 to which each can respond to each part number. Each entry contains the version number which shows the level of the maximum version of rating grant. It is shown that the version number of 0 does not have rating grant in every version of a product. A version number can be scrambled in order to strengthen further the degree of protection to the alteration of the product lock table 460.

[0037] Next, actuation of the software module on the computer system 101 by the desirable example of this invention is described. There are four parts in this actuation. The 1st actuation is arranged in the software module of the object code format that two or more rating verification triggers can be performed. The 2nd actuation generates the enciphered rating grant key 111 which permits access to a software module. Computer system 101 receives, decodes and memorizes the rating grant key 111, and the 3rd actuation sets up the product lock table 460. The 4th actuation makes a system 101 verify rating, when a software module is performed on a system 101 and it meets with a rating verification trigger. Two actuation to begin is carried out under management of a software distribution person. Two next actuation is carried out on a customer's system 101.

[0038] A software distribution person has to arrange a rating verification trigger in object code, when compiling a software module. This typical process that happens with the computer system 125 for development is shown in drawing 7 . Without including a rating verification trigger, it usually passes along a source code and it is generated by the programmer. A source code is inputted into a compiler 126 at step 701, and a program template is created at step 702. The program template includes the machine instruction of the virtual-machine level 404 (namely, level above the machine interface 405). At step 704, a program template works as an input to a translator 127 while identifying the part number and version number. Automatically, a translator 127 generates most number of rating verification triggers, inserts this in the random location in object code, and solves reference after insertion of a rating verification trigger. The rating verification trigger is contained in the software module of the object code format which is acquired as a result of being outputted at step 705 and which can be performed. The software module of the format in which this activation is possible includes the object code instruction of the executable code level 403.

[0039] The process which generates the enciphered rating grant key is shown in drawing 8 . At step 801, the order of the license over one piece or two or more

program products of a certain version level which were generated by the selling person in charge is inputted into the computer system 124 for marketing. Theoretically, although a customer is able to order the product of much version level, there is almost no reason for generally doing so. However, since each rating grant key operates only on the specified version level, if an order for the version level from which a customer differs is placed, it must generate a separate rating grant key. If an order of a customer is received, the key generation / encryption program 122 under activation will access the information about a customer, and the database 123 which specifically includes the serial number of a machine, and the information on a processor format by the system 124 at step 802. The tariff group field 201 and the machine serial number field 204 of the rating grant key 200 which are not enciphered are generated using this information. At step 803, the remaining fields are generated by an order of a customer and the reference to the database of possible part number offer, and the rating grant key of a perfect non-code format is built by them. Subsequently, key generation / encryption program 122 enciphers a rating grant key at step 804 using one of some known encryption techniques by this technical field. Subsequently, the enciphered rating grant key 111 which is step 805 and was

obtained as a result is transmitted to a customer. Although the key 111 is shown by drawing 1 as two or more binary bits, in order to simplify work which inputs a rating grant key from a keyboard, binary bits, such as an equivalent by the hexadecimal digit or the alphabetic character, may be shown to a customer in a certain format which carried out grouping.

[0040] The process in which reception and the product lock table 460 are maintained for the rating grant key 111 on computer system 101 is shown in drawing 9 a and drawing 9 b. At step 901, a customer inputs the rating grant key 111 into computer system 101 through a console 109. When this is initial installation, the install input routine 440 converses with an operator, and receives an input. When that is not right, the general input routine 441 receives an input. A rating grant key is passed to the lock discharge routine 430 which processes a decoding process. The lock discharge routine 430 makes the machine-key acquisition function 420 search the machine serial number with step 902, and makes it generate a machine key at it. Subsequently, the lock discharge routine 430 decodes the rating grant key 111 at step 903 using a machine key. Subsequently, at step 904, the product key table 450 on which the lock discharge routine 430 was coded is reconstructed so that it may state below.

The decoded rating grant key takes the form shown in drawing 2 . 80-bit product rating grant Flagg Alley 205 who shows whether this is having the lock canceled under a rating grant key with the new product for every part number is included. It is considered that a new rating grant key is the permuted key to all the products of which it cancels a lock. The lock discharge routine 430 scans each product rating grant Flagg 205 in the decoded rating grant key (step 904). When product rating grant Flagg is set to "1" (those with rating grant are pointed out) at step 905, the entry to which it corresponds in the product key table 450 at step 908 is permuted with a new rating grant key, a version number, and a tariff group value. The rating grant bit field 506 are set to "1", and the stamp bit field 503 are set to the value of the zero which show that the rating grant key is not used yet at the time of day/. When product rating grant Flagg of a new key is "0", unless a version number and the tariff group number are the same as what is memorized by the product key table 450, a new rating grant key is ineffective. When a version number and the tariff group number are the same (step 906), a rating grant key has the effectiveness which locks a product. Therefore, it is step 907, and in order to set the version number in the product lock table 460 to "0", the lock discharge routine 430 calls the lock setting up function 421, it is step 908

and permutes the entry to which it corresponds in the product key table 450 with the value of a new rating grant key. When the product key table 450 is reconstructed, it is step 909 and the contents are saved to a store.

[0041] Unless the product to which rating was given before loses rating, reconstruction of the product key table 450 does not have direct effect on the product lock table 460. A product has a lock canceled according to a demand. An exception will be generated by the system, if it meets with a rating verification trigger when performing first the software product to which rating is not given in front. Subsequently, at step 920, in order that the exception-handling routine 432 may try lock discharge of a product, the lock discharge routine 430 is called. Next, the rating grant key enciphered from the suitable entry in the product key table 450 on which the lock discharge routine 430 was coded at step 921 is taken out, a machine key is obtained at step 922, and a rating grant key is decoded at step 923. When rating is given (step 924), it is step 926, and in order to set up a version number in the entry 601 of the product lock table 460 corresponding to the part number of a software product, the lock setting up function 421 is called. The lock discharge routine 430 records the time of the 1st use on the stamp field 503 at step 927 at coincidence at the time of day/.

Subsequently, at step 928, a rating verification trigger is retried and it continues program execution. When rating grant is not shown by step 924, it is step 925 and program execution is closed.

[0042] The product lock table 460 is memorized by RAM and does not remain after the reinitialization of a system. During reinitialization ("IPL"), in order to reconstruct the product lock table 430, the IPL lock re-verification routine 431 is called. As mentioned above, this routine verifies rating, in order to reconstruct the entry to which it corresponds in the product lock table 460, it obtains a machine key, and it decodes systematically each rating grant key entry in the coded product key table 450.

[0043] The process by the desirable example in which a software module is performed is shown in drawing 10 . Activation of the software module by the system 101 is made by what this is taken out and performed for (step 1002) (step 1001) until a modular object code instruction is completed (step 1003). When an instruction is the rating verification trigger 301 (step 1004), the lock checking feature 422 is called. At step 1005, the lock checking feature 422 accesses the product lock table entry 601 corresponding to the part number contained in a rating verification trigger. Rating for it being equal to the version

number 303 by which the version number in the product lock table 460 is contained in the rating verification trigger 301, or performing software, in being larger than it is given (step 1006). In this case, the lock checking feature 422 does not perform treatment beyond it, but a system progresses to activation of the next object code instruction in a software module. When rating is not given to software, a lock checking feature generates exception condition and delivery and the exception-handling routine 432 make the exception-handling routine 432 end program execution for control (step 1007). A system does not save the result of rating inspection which shows that rating is given to software. Therefore, if it meets with a rating verification trigger again in a software module, a system will verify rating again as mentioned above.

[0044] It is possible for another example to define an addition so that a rating verification trigger can be poured in into object code through the compile path of a conventional type. This should serve as a bit of the instruction in which the activation as the module of an object code format being performed by the system with the same machine interface which a customer and a compiler implementer can use is possible. or [making a rating verification trigger into an invalid, in order that a format of object code may support the compile path of the

conventional type known by the customer] – or it may become suitable to add the obstruction over "patching" of object code which is changed. One of such the additional obstructions is defining a rating verification trigger, as other functions of a certain are achieved to coincidence. In this case, it is important that the alternate function carried out by the rating verification trigger cannot carry out with other simple instructions. This alternate function must be selected so that some instructions which achieve that function quite certainly [any compiled software modules] may be included. When having agreed in these criteria, a compiler can generate automatically the object code to which the alternate function is achieved (also in case of it, simultaneously rating verification trigger) as a part of the usual compile procedure. This definition should bring about the important obstruction over "patching" of object code which makes a rating verification trigger an invalid. Other examples are defining it as having to arrange to the addressable position which has simple relation to the part number from which it discriminates a rating verification trigger in object code. It is comparatively easy **** that a compiler pours these instructions into a suitable code position as a part of usual compile process, and easy **** performs this additional verification in the instruction embodiment. This definition should serve

as an important obstruction over patching of object code which changes discernment of the part number supplied in a rating verification trigger.

[0045] The desirable example supports 80 independent part numbers. Please understand that the actual number of the part numbers supported by this invention is adjustable. In the desirable example, it is also expected that the number of the software modules which can be compiled separately distributed by the distribution person exceeds 80 sharply. The number of part numbers is equivalent to the number of the software packages which are offered by a distribution person's marketing organization and by which a price setup was carried out separately. Although each software module has only one part number, many software modules which share this part number may exist. For example, a distribution person offers the WORD processing package containing the separate software module treating editing on screen, spell checking, document formatting, etc. When such a software module can always license as some WORD processing packages, those modules will have a common part number. It is also possible to have a separate number for every software module in the another example.

[0046] A surcharge for software to maintain upgrade, although licensed at a

lump sum payment tariff is able to charge in the desirable example. In the another example, the license of the software between a certain periods is permissible. In such an another example, a rating grant key will include the field of the addition which shows the die length of the period when software is licensed. In order to judge whether the license completed the stamp as compared with the die length of the period of license consent at the time of day/relevant to the part number in the coded product key table 450, the IPL lock re-verification routine 431 is called periodically. When a license expires at this time, activation beyond it of a software module can be prevented until new rating grant is obtained by preparing the lock discharge bit (not shown) in the entry to which it corresponds in the product lock table 460, and setting this to "0."

[Effect of the Invention] The approach and equipment which manage use of the software in computer system by this invention and which have been improved are offered.

DESCRIPTION OF DRAWINGS

[Brief Description of the Drawings]

[Drawing 1] It is drawing showing the main elements of the software protection feature by the desirable example of this invention.

[Drawing 2] It is drawing showing the contents as which the rating grant key by the desirable example of this invention is not enciphered.

[Drawing 3] It is drawing showing the contents of the typical software module by the desirable example which can be performed.

[Drawing 4] It is drawing showing the structure of hardware required on a customer's computer system in order to support the software protection feature by the desirable example, and software.

[Drawing 5] It is drawing showing a format of the coded product key table by the desirable example.

[Drawing 6] It is drawing showing a format of the product lock table by the desirable example.

[Drawing 7] It is the block diagram of a step required to arrange the rating verification trigger by the desirable example to a software module.

[Drawing 8] It is the block diagram of a step required to generate the enciphered rating grant key according to a desirable example.

[Drawing 9] It is the block diagram of a step required to decode a rating grant key

on a customer's computer system by the desirable example, and maintain record of a rating grant situation.

[Drawing 10] It is the block diagram of a step required to verify rating during the activation of a software module by the desirable example.

[Description of Notations]

101 Customer Side Computer System

102 Arithmetic and Program Control (CPU)

103 Control Storage

104 Random Access Memory (RAM)

106 Storage

107 Storage

108 Storage

109 Console

110 Medium Reader

112 Software Record Medium

120 Warehouse

121 Selling Office

123 Data Base

124 Computer System for Marketing

125 Computer System for Development

126 Compiler

127 Translator

(19)日本国特許庁(JP)

(12) 公開特許公報(A)

(11)特許出願公開番号

特開平5-334072

(43)公開日 平成5年(1993)12月17日

(51)Int.Cl.⁵

G 0 6 F 9/06

識別記号

庁内整理番号

F I

技術表示箇所

4 5 0 C 7232-5B

審査請求 有 請求項の数26(全 20 頁)

(21)出願番号 特願平3-308350

(22)出願日 平成3年(1991)10月29日

(31)優先権主張番号 6 2 9 2 9 5

(32)優先日 1990年12月14日

(33)優先権主張国 米国(US)

(71)出願人 390009531

インターナショナル・ビジネス・マシーンズ・コーポレーション

INTERNATIONAL BUSINESS MACHINES CORPORATION

アメリカ合衆国10504、ニューヨーク州アーモンク(番地なし)

(72)発明者 ロバート・カール・ビーチャー

アメリカ合衆国55904、ミネソタ州ロチェスター、17番ストリート、サウス・イースト 910番地

(74)代理人 弁理士 頓宮 孝一(外4名)

最終頁に続く

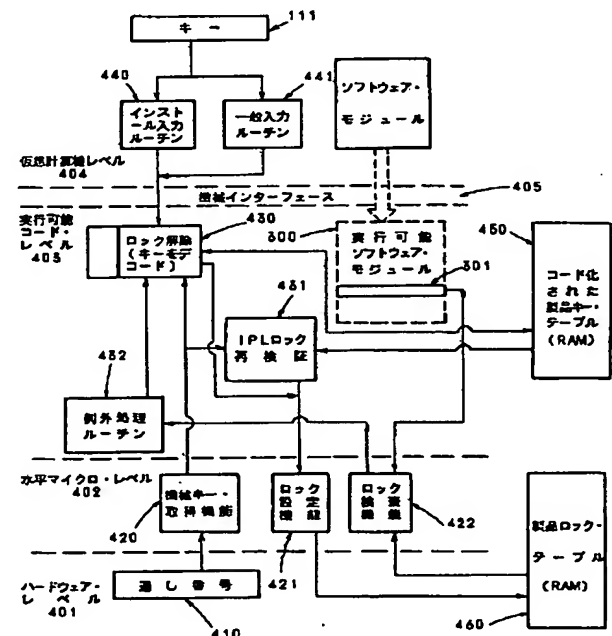
(54)【発明の名称】 ソフトウェアの使用を管理するための装置及び方法

(57)【要約】

【目的】本発明の目的は、コンピュータ・システムにおけるソフトウェアの使用を管理する改善された方法及び装置を提供することにある。

【構成】ソフトウェアは、実行する資格付与なしで配布され、別途配布される暗号化された資格付与キーによって、はじめてそのソフトウェアの実行が可能になる。このキーは、ソフトウェアのライセンスが与えられるコンピュータの通し番号と、どのソフトウェア・モジュールが機械上で走行する資格を付与されているかを示す、複数の資格付与ビットを含んでいる。配布されたソフトウェアは、複数の資格検証トリガを含む。各トリガは、ソフトウェア・モジュールの製品番号を識別する、目的コード形式の単一の機械語命令である。

【効果】本発明によって、コンピュータ・システムにおけるソフトウェアの使用を管理する、改善された方法及び装置が提供される。



【特許請求の範囲】

【請求項1】 コンピュータ・システムにソフトウェア・モジュールを実行する資格を付与する手段と、上記ソフトウェア・モジュール内に配置された、資格の検証をトリガする複数の独立したトリガ手段と、上記の複数の独立したトリガ手段のそれぞれにตอบสนองして、上記コンピュータ・システムが上記ソフトウェア・モジュールを実行する資格をもつことを検証する資格検証手段と、

上記の資格検証手段にตอบสนองして、上記コンピュータ・システムが上記のソフトウェア・モジュールを実行する資格をもたないと上記資格検証手段が判定した場合に、上記ソフトウェア・モジュールの実行を打ち切る手段とを備える、コンピュータ・システムで実行されるソフトウェア・モジュールの使用を管理するための装置。

【請求項2】 上記の複数の独立したトリガ手段が、上記資格検証手段をトリガする単一の目的コード命令である、請求項1に記載の、ソフトウェア・モジュールの使用を管理するための装置。

【請求項3】 上記資格検証手段をトリガする上記目的コード命令が、上記ソフトウェア・モジュールの適切な実行に必要な追加ステップをも実行し、その結果、上記目的コード命令を除去することによって上記ソフトウェア・モジュールが修正される場合は、上記の追加ステップが実行されず、上記ソフトウェア・モジュールが適切に実行されなくなる、請求項2に記載の、ソフトウェア・モジュールの使用を管理するための装置。

【請求項4】 上記資格検証手段をトリガする上記目的コード命令が、上記のソフトウェア・モジュールの製品番号を含む、請求項2に記載の、ソフトウェア・モジュールの使用を管理するための装置。

【請求項5】 上記資格検証手段が、資格付与情報を含みかつそれぞれが製品番号に関連する複数のエントリを備える、上記コンピュータ・システム内の製品ロック・テーブルと、上記目的コード命令にตอบสนองして、上記目的コード命令に含まれる上記製品番号に関連する上記製品ロック・テーブル中のエントリに含まれる上記資格付与情報にアクセスする手段とを含む、請求項4に記載の、ソフトウェア・モジュールの使用を管理するための装置。

【請求項6】 上記のコンピュータ・システムに上記のソフトウェア・モジュールを実行する資格を付与する上記手段が、

データから構成される資格付与キーを生成する手段と、上記の資格付与キーを上記コンピュータ・システムに入力する手段とを備える、請求項1に記載の、ソフトウェア・モジュールの使用を管理するための装置。

【請求項7】 上記コンピュータ・システムが独自の識別子を含み、

資格付与キーを生成する上記手段が、上記の独自の識別

子を用いて上記資格付与キーを生成し、

上記資格付与キーが、同じ独自の識別子を含むコンピュータ・システム上でのみソフトウェアを実行する資格を与える、

請求項6に記載の、ソフトウェア・モジュールの使用を管理するための装置。

【請求項8】 上記の独自の識別子を用いて上記資格付与キーを暗号化し、その結果得られる暗号化された資格付与キーが同じ独自の識別子をもつコンピュータ・システム上でしか解読できないようにする手段と、

上記コンピュータ・システムの独自の識別子にアクセスする、コンピュータ・システム内の手段と、

上記の独自の識別子にアクセスする上記手段にตอบสนองして、上記の暗号化された資格付与キーを解読する、上記のコンピュータ・システム内の手段とを備える、請求項7に記載の、ソフトウェア・モジュールの使用を管理するための装置。

【請求項9】 資格の検証をトリガする複数の独立したトリガ手段をソフトウェア・モジュール内に配置する段階と、

上記ソフトウェア・モジュールを実行する段階と、

上記ソフトウェア・モジュールの実行中に上記複数の独立したトリガ手段の一つに会ったとき、コンピュータ・システムがそのソフトウェア・モジュールを実行する資格をもつことを検証する資格検証動作を上記コンピュータ・システム中でトリガする段階と、

上記資格検証で、上記コンピュータ・システムには上記ソフトウェア・モジュールを実行する資格がないと判定された場合、上記ソフトウェア・モジュールの実行を打ち切る段階とを含む、コンピュータ・システム上で実行されるソフトウェア・モジュールの使用を管理するための方法。

【請求項10】 複数の独立したトリガ手段を上記ソフトウェア・モジュール内に配置する上記段階が、上記ソフトウェア・モジュール内の複数の別々の位置に、上記資格検証動作をトリガする単一の目的コード命令を配置することを含む、請求項9に記載の、ソフトウェア・モジュールの使用を管理するための方法。

【請求項11】 上記資格検証動作をトリガする上記目的コード命令が、上記ソフトウェア・モジュールの適切な実行に必要な追加ステップをも実行し、その結果、目的コード命令を除去することによって上記ソフトウェア・モジュールが修正される場合は、上記の追加ステップが実行されず、上記ソフトウェア・モジュールが適切に実行されなくなる、請求項10に記載の、ソフトウェア・モジュールの使用を管理するための方法。

【請求項12】 上記資格検証動作をトリガする上記目的コード命令が、上記ソフトウェア・モジュールの製品番号を含む、請求項10に記載の、ソフトウェア・モジュールの使用を管理するための方法。

【請求項13】資格付与情報を含みかつそれぞれが製品番号に関連する複数のエントリを備えた製品ロック・テーブルを、上記のコンピュータ・システム内で維持する段階を含み、資格検証動作をトリガする上記段階が、上記目的コード命令中に含まれる製品番号に関連する製品ロック・テーブル中のエントリに含まれる上記資格付与情報にアクセスすることを含む、請求項12に記載の、ソフトウェア・モジュールの使用を管理するための方法。

【請求項14】複数のデータ・ビットから構成され、上記コンピュータ・システムが上記ソフトウェア・モジュールを実行する資格をもつかどうかを判定できるようにする情報を提供する資格付与キーを生成する段階と、上記資格付与キーを上記コンピュータ・システムに入力する段階とを含む、請求項9に記載の、ソフトウェア・モジュールの使用を管理するための方法。

【請求項15】上記コンピュータ・システムが独自の識別子を含み、資格付与キーを生成する上記段階が、上記の独自の識別子を用いて上記資格付与キーを生成し、上記資格付与キーが、同じ独自の識別子を含むコンピュータ・システム上でのみソフトウェアを実行する資格を与える、請求項14に記載の、ソフトウェア・モジュールの使用を管理するための方法。

【請求項16】ソフトウェア・モジュールを実行する資格を受け取る手段を有し、かつ上記ソフトウェア・モジュール内のトリガ手段にตอบสนองして、上記コンピュータ・システムが上記ソフトウェア・モジュールを実行する資格をもつことを検証する資格検証手段を有するコンピュータ・システム上で、上記ソフトウェア・モジュールが実行される、資格の付与を管理するためのプログラム製品であって、記録媒体上に記録された少なくとも一個のソフトウェア・モジュールと、コンピュータ・システム上で資格の検証をトリガする、上記ソフトウェア・モジュール中の複数の独立したトリガ手段とを備える、プログラム製品。

【請求項17】上記複数の独立したトリガ手段のそれぞれが、上記資格の検証をトリガする単一の目的コード命令である、請求項16に記載の、資格付与を管理するためのプログラム製品。

【請求項18】上記資格検証手段をトリガする上記目的コード命令が、上記ソフトウェア・モジュールの適切な実行に必要な追加ステップをも実行し、その結果、上記目的コード命令を除去することによって上記ソフトウェア・モジュールが修正される場合は、上記追加ステップが実行されず、ソフトウェア・モジュールが適切に実行されなくなる、請求項17に記載の、資格付与を管理するためのプログラム製品。

【請求項19】上記資格検証手段をトリガする上記目的コード命令が、上記ソフトウェア・モジュールの製品番号を含み、

上記コンピュータ・システム上の上記資格検証手段が、資格付与情報を含みかつそれぞれが製品番号に関連する複数のエントリを備えた製品ロック・テーブルをコンピュータ・システム内に有する、請求項17に記載の、資格付与を管理するためのプログラム製品。

【請求項20】ソフトウェア・モジュールを実行する資格を受け取る手段を有し、かつ上記ソフトウェア・モジュール内のトリガ手段にตอบสนองして、上記コンピュータ・システムが上記ソフトウェア・モジュールを実行する資格をもつことを検証する資格検証手段を有するコンピュータ・システム上で、上記ソフトウェア・モジュールが実行される、ソフトウェア・モジュールを配布する方法であって、

資格の検証をトリガする複数の独立したトリガ手段を上記ソフトウェア・モジュール内に配置する段階と、

上記ソフトウェア・モジュールを上記コンピュータ・システムに配布する段階と、

上記コンピュータ・システムに上記ソフトウェア・モジュールを実行する資格を付与する段階とを含む、ソフトウェア・モジュールを配布する方法。

【請求項21】上記複数の独立したトリガ手段を上記ソフトウェア・モジュール内に配置する上記段階が、上記資格の検証をトリガする単一の目的コード命令を、上記ソフトウェア・モジュール中の複数の別々の位置に配置することを含む、請求項20に記載の、ソフトウェア・モジュールを配布する方法。

【請求項22】上記資格検証をトリガする上記目的コード命令が、上記ソフトウェア・モジュールの適切な実行に必要な追加ステップをも実行し、その結果、上記目的コード命令を除去することによって上記ソフトウェア・モジュールが修正される場合は、上記の追加ステップが実施されず、ソフトウェア・モジュールが適切に実行されなくなる、請求項21に記載の、ソフトウェア・モジュールを配布する方法。

【請求項23】上記資格検証をトリガする上記目的コード命令が、上記ソフトウェア・モジュールの製品番号を含み、

コンピュータ・システム上の上記資格検証手段が、資格付与情報を含みかつそれぞれが製品番号に関連する複数のエントリを備えた製品ロック・テーブルを、コンピュータ・システム内に有する、

請求項21に記載の、ソフトウェア・モジュールを配布する方法。

【請求項24】上記コンピュータ・システムに上記ソフトウェア・モジュールを実行する資格を付与する上記段階が、

複数のデータ・ビットから構成され、上記コンピュータ・システムが上記ソフトウェア・モジュールを実行する資格をもつかどうかを判定できるようにする情報を提供する、資格付与キーを生成する段階と、

上記資格付与キーをコンピュータ・システムに伝送する段階とを含む、請求項20に記載の、ソフトウェア・モジュールを配布する方法。

【請求項25】上記コンピュータ・システムが独自の識別子を含み、

資格付与キーを生成する上記段階が、上記の独自の識別子を用いて上記の資格付与キーを生成し、

上記資格付与キーが、同じ独自の識別子を含むコンピュータ・システム上でのみソフトウェアを実行する資格を与える、

請求項24に記載の、ソフトウェア・モジュールを配布する方法。

【請求項26】上記ソフトウェア・モジュールを配布する上記段階が、それぞれ単一の記憶媒体上に資格検証をトリガする複数の独立したトリガ手段を有する、複数のソフトウェア・モジュールを配布することを含み、

上記コンピュータ・システムに上記ソフトウェア・モジュールを実行する資格を付与する上記段階が、上記の単一の記憶媒体上の上記複数のソフトウェア・モジュールのうち少なくとも2つに別々の資格を与えることを含む、

請求項20に記載の、ソフトウェア・モジュールを配布する方法。

【発明の詳細な説明】

【0001】

【産業上の利用分野】本発明は、コンピュータ・ソフトウェアの使用に関し、より具体的には、コンピュータ・ユーザがライセンス・ソフトウェアをライセンスに従わない形で使用できるのを制限することに関する。

【0002】

【従来の技術】現代のコンピュータ・システムは、数えきれない時間にわたる技術上及びプログラミング上の熟練の成果をその設計に取り入れた、極めて複雑な機械である。コンピュータ・システムは多くの構成要素を含んでいるが、それらはハードウェアとソフトウェアに大別できる。ハードウェアは、システムを構成する、有形の回路、ボード、ケーブル、記憶装置、格納装置などである。しかし、タイプライタでピューリツァー賞を受賞できる脚本を書けないのと同様、ハードウェアは、それ自体で、現実の世界の問題を解決することはできない。ハードウェアは、何をすべきかを告げる命令を必要とする。「ソフトウェア」とは、そのもっとも純粋な形では、ハードウェアに有用な仕事を実行させる命令を言う。(ただし漠然と、ソフトウェアが記憶され配布されている媒体に適用されることもある)。ハードウェアと同様に、ソフトウェアも人間の創意工夫の産物である。

高品質のソフトウェアは、プログラマとも呼ばれる作成者の側の、かなりの創造性、訓練、知能を必要とする。多くの大学が、人々にソフトウェアを作成する技術を教える目的で、「コンピュータ科学」及び類似の学科の教育課程を設けている。この業界全体が、何千人ものキャリアを抱え、有用な仕事をするソフトウェアを作成するまでに成長して来た。ソフトウェアの開発に莫大な労力が費やされる結果、コンピュータ・システムの一部分であるソフトウェアの価値が、ハードウェアの価値を上回ることも珍しくはない。

【0003】現代のコンピュータ・システムの特徴の一つは、非常に高速度で容易にデータを伝送し複製することである。一般的に、これは必要かつ有益な能力である。しかし、このことはまた、何年も労力をかけて作成したソフトウェアが、比較的安価な磁気媒体で1秒の何分の一かで複製でき、濫用される可能性があることを意味している。許可を得ていない人々が、僅かな費用と労力で、価値のあるソフトウェアを複製することができ。この慣行は、一般に、ソフトウェア著作権侵害と呼ばれている。近年、ソフトウェア著作権侵害を抑制するため、刑法及び民法上の義務を課する様々な法律が制定されてきている。しかしながら、ソフトウェアが比較的容易に複製でき、かつソフトウェアが高価なためにそうしたくなる誘惑にかられるため、ソフトウェアの著作権侵害は依然として問題となっている。

【0004】「パーソナル・コンピュータ」と呼ばれる小型コンピュータ用の大量配布される安価なソフトウェアの場合、全顧客に対して同額の固定した一括払い料金で、ソフトウェアの使用ライセンスを認めるのが普通である。これは、大型のメインフレーム・コンピュータ・システムでは、それほど行われていない。このような大型システム用のソフトウェアは、数百万行のコードを要することがあり、開発及び維持に非常に大量の投資を必要とする。開発者がこの投資を回収するのに充分な単一の固定した一括払い料金を設定すると、多くの小口ユーザにとって、使用できないほど高価になりがちである。したがって、メインフレームの場合は、使用量に基づくソフトウェアのライセンス料を請求するのが普通である。「段階的価格設定」と呼ばれるこのような一つの方法は、可変料金体系に従って顧客の機械の性能に基づいた料金を請求するものである。より多くの端末が接続されたより高速の機械を使用する顧客ほど、同じソフトウェアに対し、性能の劣る機械の顧客よりも高いライセンス料を支払う。もう一つの通常の慣行は、ソフトウェアの保守グレードアップに対して別途料金を請求するものである。

【0005】高品位のソフトウェアを作成するのに必要な専門知識のレベルと、それににかかる時間及び労力の量を考えると、このようなソフトウェアの作成には金がかかる。ソフトウェアの開発者がその訓練と努力に比して

報われない場合は、ソフトウェアを作成する人がいなくなる。開発者が行ったソフトウェアへの投資を保護することは、実際上の要請であるだけでなく、道徳上の要請でもある。したがって、ソフトウェアの正当な所有者は、ソフトウェアを許可なしに使用することが難しく、ソフトウェアの開発者がその製品に対して正当に報われる、ソフトウェア配布方法の開発を求めて来た。

【0006】ソフトウェアは、正当な所有者から多数の異なる方法で配布される。無許可の使用を防止する観点から、これらの配布方法は、無制限資格付与法、制限資格付与法、非資格付与法の3つのグループに大別できる。

【0007】無制限資格付与とは、配布されたソフトウェアが、その設計の対象であったどの機械でも、制限なしに走行するという意味である。無制限資格付与のソフトウェアを配布する所有者は、各ユーザに、ユーザがそれに対して対価を支払いそれを実行する資格を有するソフトウェアだけを配布しなければならない。このようなソフトウェアの受取側がこれを許可されない形で複製したり使用したりするのを妨げるものは、法的及び契約上の義務だけである。大部分のパーソナル・コンピュータで使用されているものなど、一括払い料金でライセンスが与えられる安価なソフトウェアでは、これがもっとも普通の配布方法である。

【0008】制限資格付与とは、ソフトウェアが、ユーザがそれを複写していくらでも多くの機械上で実行できるのを制限する、ある種の制限を内蔵していることを意味する。いくつかの異なる制限資格付与法がある。その一つは、配布されたオリジナルから作成できるコピーの数を制限する、複写制限である。複写制限は、ソフトウェア著作権侵害に対してあるレベルの保護を実現するが、若干の欠点をもつ。無制限資格付与と同様に、複写制限でも、所有者が各ユーザに、そのユーザが実行する資格を有するソフトウェアだけを配布することが必要となる。これはフルブーフではなく、保護機構を打ちやぶって、複写保護されたソフトウェアの文字通りのコピーをとることのできるプログラムも存在する。最後に、これは、ユーザがバックアップの目的で正当なコピーをとれる、またはソフトウェアを高速記憶装置から走行できるのを妨げる。もう一つの制限資格付与法は、ユーザまたは機械に特有の情報をソフトウェア自体の中にコード化するものである。ソフトウェアを実行する際、機械は、ソフトウェアがその機械またはユーザに許可されていることを確かめるために検査を行う。この方法は、ユーザが正当なコピーをとれるのを妨げずに、保護を実現する。ただし、これは、配布されたソフトウェアの各コピーをそれぞれ独自にコンパイルし、配布媒体に載せ、発送しなければならないので、非常に高価な配布システムが必要となる。

【0009】非資格付与法は、配布されたソフトウェア

が使用禁止にされていて、実行するには別途配布される資格付与を必要とすることを意味する。ある非資格付与法は、特注ソフトウェアの配布を完全に避ける可能性がある。ただし、必ずしもこのような方法の全てがこうした能力をもっているわけでない。たとえば、所有者は、同じ1組の多数のソフトウェア・プログラムを単一の総称媒体上でその全ての顧客に配布し、顧客が支払いを済ませたプログラムだけを実行することを許す、個別化された許可キーを各顧客に別途配布することができる。非資格付与法は、他の方法に付随する多くの問題を回避するが、現在の設計は、資格付与の偽造や著しい性能劣化を受ける恐れが大きい。大抵の場合、ソフトウェアを実行するための資格付与を差し止めるために用いられる機構は、性能劣化という検証上のオーバーヘッドを避けるため、資格検証手段をソフトウェア・モジュール内に（データあるいは命令として）集中することを必要とする。場合によっては、この資格付与のオーバーヘッドが、製品識別子のサイズ、及び配布されたソフトウェア内の保護ルーチンの実装によることがある。あるいはまた、そのオーバーヘッドがソフトウェアを走行させながら複雑な復号手段を実行する必要があることもある。こうした資格検査の集中の結果、経験あるプログラマが、「パッチング」すなわち目的コードの選択された小部分を無効にすることによって、保護機構を無効にするのは、比較的容易である。別の場合、目的コードは資格検証を行わず、モジュールを、それからビット署名を作成することによって識別する、安全呼出経路がそれを行う。これはパッチングを受け難くするが、不可避免的に呼出し機構の重大な性能劣化をひき起こす。

【0010】従来技術で教示されている保護方法は、保護レベルと使いやすさとの折合いをつけるものである。機械特有の情報をソフトウェア中にコード化することによって比較的高レベルの保護を得ることが可能であるが、配布される各ソフトウェア・コピーが独自のものとなる、非常に複雑な配布システムを維持するという犠牲を払わなければならない。より安価な配布も可能であるが、保護が一部失われるという犠牲を払わなければならない。高レベルの保護を実現し、大量配布技法を用いて容易に配布でき、かつシステムの性能、ユーザが正当にバックアップ複写用コピーをとる可能性、その他の必要な機能を不当に妨げない方法が求められている。同時に、段階的価格設定、及び異なるソフトウェアの異なるバージョンごとに別々のライセンス料をサポートする方法も求められている。

【0011】

【発明が解決しようとする課題】本発明の目的は、コンピュータ・システムにおけるソフトウェアの使用を管理する改善された方法及び装置を提供することにある。

【0012】本発明の他の目的は、コンピュータ・システムにおける無許可のソフトウェア使用に対するより高

レベルの保護を提供することにある。

【0013】本発明の他の目的は、ソフトウェアを無許可使用に対して保護する費用を削減することにある。

【0014】本発明の他の目的は、無許可使用に対して保護されたソフトウェアを実行するコンピュータ・システムの性能を増強することにある。

【0015】本発明の他の目的は、無許可使用に対して保護されたソフトウェアの配布者の配布システムを簡単にすることにある。

【0016】本発明の他の目的は、このような保護がソフトウェアの正当な使用に及ぼす影響を削減する、ソフトウェアを無許可使用から保護する方法及び装置を提供することにある。

【0017】本発明の他の目的は、ユーザがソフトウェアの正当なバックアップ用コピーをとるのを許しながら、ソフトウェアを無許可使用から保護する改善された方法及び装置を提供することにある。

【0018】本発明の他の目的は、ソフトウェアを無許可の使用が可能となるように改変するのを難しくすることにある。

【0019】本発明の他の目的は、段階的に価格設定したソフトウェアを配布する、改善された方法及び装置を提供することにある。

【0020】

【課題を解決するための手段】本発明によれば、ソフトウェアは、実行するための資格付与なしに配布される。別々に配布される暗号化された資格付与キーにより、ソフトウェアの実行が可能となる。この資格付与キーは、それに対してソフトウェアのライセンスが与えられている機械の通し番号と、どのソフトウェア・モジュールがその機械で走行する資格をもつかを指示する複数の資格付与ビットを含んでいる。安全解読機構が機械に内蔵されている。安全解読機構が機械の通し番号を取り出し、これを資格付与キーを解読するためのキーとして使用する。次いで、資格付与情報が、メモリ内の製品ロック・テーブルに記憶される。

【0021】配布されたソフトウェアは、複数の資格検証トリガを含んでいる。好ましい実施例では、各トリガは、ソフトウェア・モジュールの製品番号を識別する、目的コード中の単一機械語命令である。ソフトウェア・モジュールの実行中にこのような資格検証トリガに出会うと、機械は、ソフトウェア・モジュールの製品番号に対応する製品ロック・テーブルのエントリを検査する。製品が走行する資格をもつ場合、正常に実行が継続され、そうでない場合は、実行が打ち切られる。この検証はただ1個の機械語命令しか要しないので、システム全体の性能にほとんど影響を与えずに実行できる。その結果、かなりの数のこのような資格検証トリガを目的コード中に配置し、誰かがこのトリガを「パッチング」することによってコードを改変することを事実上不可能にす

ることができる。別の実施例では、資格検証トリガが、ソフトウェア・モジュールの適正な実行に必要な何らかの有用な仕事をも行う。これによって、ソフトウェアを「パッチング」するのが一層難しくなり、このような検証トリガの性能に対する影響がさらに軽減される。

【0022】ソフトウェア自身はどのような資格付与も含んでいないので、配布の制限は必要でない。好ましい実施例では、ソフトウェアの配布者は、複数のソフトウェア・モジュールを単一の総称媒体に記録し、記録された同じ1組のモジュールをその全ての顧客に配布することができる。各顧客は、自分がライセンスを与えられているソフトウェア・モジュールだけを実行できる、独自の資格付与キーを受け取る。顧客に、ライセンスが与えられていない何らかのモジュールが与えられても、適切なキーなしにはそれが実行できないので、大きな問題にはならない。顧客は、自由にソフトウェアを自分のシステムのその他の記憶装置にロードし、あるいはソフトウェアのバックアップ用コピーをいくらかでも作成することができる。

20 【0023】

【実施例】本発明の好ましい実施例にもとづくソフトウェア保護機構の主要要素の説明図を図1に示す。顧客のコンピュータ・システム101は、制御記憶機構103に緊密に結合された中央演算処理装置(CPU)102、ランダム・アクセス・システム・メモリ104、消去不能で電子的に読取り可能な独自の識別子105、複数の記憶装置106、107、108を含んでいる。好ましい実施例では、独自の識別子105は機械の通し番号である。好ましい実施例では、記憶装置106~108は、回転式磁気ディスク記憶装置であるが、他の記憶技術を使用することも可能である。コンピュータ・システム101はまた、オペレータからの入力を受け取る操作卓109及びソフトウェア媒体読取装置110をも含んでいる。図1には1台の操作卓、1個のソフトウェア媒体読取装置、3個の記憶装置が示されているが、システム101に取り付けられるこのような装置の実際の数は可変であることを理解されたい。また、追加の装置をシステム101に取り付けられることを理解されたい。好ましい実施例では、コンピュータ・システム101は、IBM社のAS/400コンピュータ・システムであるが、他のコンピュータ・システムも使用できる。

【0024】ソフトウェア・モジュールは、それを動作させるのに必要な資格付与キー(entitlement key)とは別に配布される。本来、ソフトウェア・モジュールは、開発用コンピュータ・システム125上で作成される。開発用コンピュータ・システム125は、コンパイラ126及びトランスレータ127を含んでいる。ソフトウェア・モジュールはソフトウェア記録媒体112上に記録され、倉庫120に格納され、そこから顧客に配布される。配布者の販売事務所121か

ら、暗号化された資格付与キー111が配布される。販売事務所121は、マーケティング用コンピュータ・システム124にアクセスすることができる。マーケティング用コンピュータ・システム124は、資格付与キーの生成/暗号化プログラム122と、顧客情報を含むデータ・ベース123とを含んでいる。具体的には、データ・ベース123は、暗号化された資格付与キーの生成に必要な機械の通し番号とプロセッサの種類の情報を含んでいる。好ましい実施例では、マーケティング用コンピュータ・システム124は、中央に位置して複数の販売事務所と通信するコンピュータである。ただし、マーケティング用コンピュータ・システム124は、販売事務所自体に位置し、局所または中央のデータ・ベースにアクセスすることもできる。図1には、配布者の制御下にある別々の2つのコンピュータ・システム124、125が示されているが、両方の機能を実行する物理的に単一のコンピュータ・システムでもよいことを理解されたい。

【0025】ソフトウェアの配布者から顧客に、暗号化された資格付与キー111が郵送、電話またはその他の適当な手段で送られる。資格付与キーを電子的にまたはフロッピー・ディスクなどの磁気媒体上で伝送することが可能であるが、キーは、オペレータが操作卓109上でタイプして、これをシステム101中に入力できるほど十分簡潔なものである。

【0026】好ましい実施例では、多数のソフトウェア・モジュールが同一媒体112上で配布される。配布者は、資格付与キーを介して各モジュールを使用する資格を独立に与えることができる。倉庫120は、媒体112上に、総称的に1組のソフトウェア・モジュールを格納する。各顧客に、どのモジュールを使用するライセンスが与えられているかにかかわらず、同じ総称的な1組のソフトウェア・モジュールが発送される。別途に配布される資格付与キーは、どのソフトウェア・モジュールをその上で実行する資格があるかをシステム101が決定するための情報を含んでいる。

【0027】好ましい実施例では、ソフトウェア媒体112は、一枚または複数の読取り専用光ディスクから構成され、媒体読取装置110は光ディスク読取装置である。ただし、電子式配布媒体やその他の配布媒体も使用できることを理解されたい。ソフトウェア媒体112を受け取ると、通常、顧客は、所望のソフトウェア・モジュールを媒体読取装置110からシステム101にロードし、そのソフトウェア・モジュールを記憶装置106~108に記憶する。顧客は、ソフトウェア・モジュールの一つまたは複数のバックアップ用コピーを任意の適当な媒体上に生成し、これらをアーカイブに記憶することができる。ソフトウェア媒体112は、システムへの複写やロードに対する制限を含んでおらず、自由に複写可能かつロード可能である。

【0028】好ましい実施例による暗号化前の資格付与キー200の内容が図2に示されている。このキーは、料金グループ・フィールド201、ソフトウェア・バージョン・フィールド202、キー形式フィールド203、機械通し番号フィールド204及び製品資格付与フラグ205を含んでいる。料金グループ・フィールド201は、16通りの機械段階値から、1つを指定し、ソフトウェアの段階的価格設定をサポートするのに使用される。ソフトウェア・バージョン・フィールド202は、資格が与えられるソフトウェアのバージョン・レベルを指定する。ソフトウェアのグレードアップを維持するための別途料金が課せられることが予想されている。資格付与キー200中で指定されるバージョンは、そのバージョン・レベルより以前（下位）の全レベルのソフトウェアに資格を与える。キー形式フィールド203はキー・フォーマット、キーの関連性の将来変更のため、あるいはサポートされる異なる製品の番号拡張のために保留された領域である。機械通し番号フィールド204は、資格付与キーが対象としている機械の通し番号を含んでいる。製品資格付与フラグ205は、それぞれが製品番号に対応する80本の別々の製品フラグを含む、80ビットのフィールドである。対応する製品番号に資格が与えられている場合、このビットは“1”にセットされ、そうでない場合は、“0”にセットされる。

【0029】好ましい実施例では、ソフトウェア・モジュールは、コンパイルされた目的コードとして配布される。典型的なソフトウェア・モジュール300を図3に示す。このソフトウェア・モジュールは、コンピュータ・システム101上で実行できる複数の目的コード命令を含んでいる。本発明によれば、いくつかの資格検証トリガ命令（以下、「資格検証トリガ」と表記）301が目的コードに組み込まれている。ソフトウェア・モジュール内に含まれる全ての資格検証トリガ301は、同一である。資格検証トリガ301は、命令コード・フィールド302、バージョン・フィールド303、製品番号フィールド304を含んでいる。フィールド305は使用されていない。命令コード・フィールド302は、実行すべき操作を識別する、目的コード命令の動詞部分である。バージョン・フィールド303は、ソフトウェア・モジュールのバージョン・レベルを識別する。製品番号フィールド304は、そのソフトウェア・モジュールに関連する製品番号を識別する。バージョン及び製品番号の情報はモジュールごとに変わるが、全ての資格検証トリガには単一の命令コードが使用される。別の実施例では、資格検証トリガは、他の何らかの有用な仕事を実行する直接命令（とりわけ、その処理用のオペランドを命令中で指定しておく必要のない命令）でもある。このような別の実施例では、トリガ命令が実行されると、システム101は資格検証と同時に他の何らかの操作を実行する。

【0030】コンピュータ・システム101は、暗号化された資格付与キー111を受け取ってデコードする手段、ならびに資格検証トリガに応答してシステムがソフトウェア・モジュールを実行する資格をもつことを検証する手段を含んでいる。好ましい実施例では、図4に示されているように、これらの機能は、異なるレベルのシステム・ハードウェア及びシステム・ソフトウェアの間で分割されている。この実施例では、システム101は、ハードウェア・レベル401、水平マイクロコード・レベル402、実行可能コード・レベル403、仮想

計算機レベル404という、4レベルのハードウェア/ソフトウェア機能を含んでいる。機械インターフェース405が、仮想計算機レベルを下位の全てのレベルから分離している。機械インターフェース405は、顧客に対して定義されている最も下位レベルにあるインタフェースである。すなわち、仮想計算機レベルの命令ビットは顧客に対して定義されるが、それより下のレベルの操作は定義されない。したがって、顧客は、機械インターフェース・レベルより下のレベルの命令を直接変更する能力がない。永久的な、変更不能な機械通し番号410が、ハードウェア・レベル401に格納されている。

【0031】水平マイクロコード402は、実行可能な命令セットを解釈するマイクロコード・エントリを含んでいる。これは、制御記憶機構103、すなわち好ましい実施例では、顧客による変更が不可能な読取り専用メモリ（ROM）に、物理的に記憶されている。水平マイクロコード中のエントリは、機械キー取得機能420、ロック設定機能421、ロック検査機能422をサポートしている。機械キー取得機能420は、好ましい実施例ではシステム通し番号に基づく、独自の識別子を、ある永久的なハードウェア位置から取り出す。ロック設定機能421は、製品ロック・テーブル460にアクセスし、テーブル中のエントリを変更する。ロック設定機能は、製品ロック・テーブル460を変更できる唯一のマイクロコード機能である。ロック検査機能422は、製品ロック・テーブル460にアクセスし、エントリの一つを読み取って資格が付与されているかどうかを検証する。

【0032】実行可能コード・レベル403は、下位レベルの監視サポート、及び機械インターフェースで定義された命令を実施するサポートを含んでいる。これはメモリに物理的に記憶されるが、その内部仕様が顧客に対して定義されていないので、実際上は顧客によって変更できない。下位レベルのサポートには、ロック解除ルーチン430、初期プログラムロード（IPL）ロック再検証ルーチン431、例外処理ルーチン432が含まれる。アンロック・ルーチン430は、固有の機械キーを用いて資格付与キー111をデコードし、暗号化された資格付与キー111を製品キー・テーブル450に記憶する。初期プログラムロードロック再検証ルーチン43

1は、システムが再び初期設定されると、コード化された製品キー・テーブル450の内容を取り出して、製品ロック・テーブル460を再構築する。例外処理ルーチン432は、水平マイクロコード・レベル402の機能によって生成される例外条件に응答する。実行可能コード・レベル403は、目的コード形式のソフトウェア・モジュール300を含んでいる。

【0033】仮想計算機レベル404は、機械語命令によって表されているものとしてソフトウェアを参照する。このレベルの計算機は、機械語命令が直接実行可能ではないという意味で、仮想計算機として動作する。仮想計算機レベル404は顧客に利用できる最も下位レベルのインターフェースなので、システム上で実行可能な目的コード形式のモジュールを作成するのに、非従来型のコンパイル経路が必要である。このコンパイル処理については、後に説明する。厳密には、この非従来型のコンパイル経路を介して作成されるソフトウェアは、実行しようとする実行可能な目的コードの形式をとらなければならないが、機械語命令が直接実行可能であるかのように、仮想計算機レベルの機械語命令で表して議論するのが普通であり、よりわかりやすい。これが確立されると、仮想計算機レベル404がコンパイルしたユーザのソフトウェアを含んでいると言うことができる。これはまた、システム101に対するより上位レベルのオペレーティング・システムのサポートも含んでいる。仮想計算機レベル404でこのオペレーション・システムのサポートは、資格付与キーの入力をサポートするのに必要な2個のユーザ・インターフェース・ルーチンを含んでいる。一般入力ルーチン441は、通常の操作中に入力を処理するのに使用する。さらに、オペレーティング・システムの初期導入中に資格付与キーを入力するには特別のインストール入力ルーチン440が必要である。これが必要なのは、機械インターフェース・レベル405より上位のオペレーティング・システムの部分が、本発明では他のプログラム製品として扱われるためである。すなわち、このような部分は製品番号をもち、その目的コードは資格検証トリガの影響を受ける。インストール入力ルーチン440は、オペレーティング・システムの、資格検証トリガをもたない唯一の部分であり、従って、システムを最初に導入する際に資格付与キーが入力できるようになる。

【0034】ソフトウェア・モジュール300は、システム101上で実行されるコンパイルされた目的コード形式のプログラム製品の一部である。これは、仮想計算機レベルの他の対象にアクセス可能であるという意味で、仮想計算機レベル404のエンティティとして存在する。ただし、実際に実行可能なコードは、破線の枠で示されているように、実行可能コード・レベル403で動作する。実行可能コードは、水平マイクロコードのロック検査機能422によって実行される、資格検証トリ

が301（一つだけが図に示されている）を含んでいる。

【0035】コード化された製品キー・テーブル450が図5に示されている。製品キー・テーブル450はランダム・アクセス・メモリ104に入っており、不揮発性記憶装置に複製されているため、システムの電力を遮断したり、その他の理由で再初期設定しなければならない場合に、回復することが可能である。テーブル450は、それぞれが各製品番号に対応し得る80個のエントリ501を含んでいる。各エントリ501は、そのエントリの製品番号に適用される暗号化された資格付与キー502と、そのキーが最初にいつ使用されたかを示す日時スタンプ503と、そのキーのバージョン番号504と、そのキーの料金グループ505と、そのキーが製品をアンロックするかどうかを示す資格付与ビット506との完全なコピーを含んでいる。日時スタンプ503は暗号化することができる。バージョン番号504、料金グループ505及び資格付与ビット506は、暗号化された資格付与キー502に含まれる情報を繰り返す。それらは、照合をサポートするためにこのテーブルに含まれている。ただし、最初に暗号化された資格付与キー502中の情報を検証してからでないと、製品ロック・テーブル460のエントリを改変して、プログラム実行の資格を与えることはできない。製品キー・テーブル450中の各資格付与キー502は暗号化された形なので、ユーザーのこのテーブルへのアクセスを防止するために特別の措置は必要でない。

【0036】図6に、製品ロック・テーブル460が示されている。このテーブルは、水平マイクロコードの完全な制御下にある、ランダム・アクセス・メモリ104の特別な下位アドレス範囲に入っている。製品ロック・テーブル460は、それぞれが各製品番号に対応し得る80個のエントリ601を含んでいる。各エントリは、資格付与の最大バージョンのレベルを示すバージョン番号を含んでいる。0のバージョン番号は、製品のどのバージョンにも資格付与がないことを示している。製品ロック・テーブル460の改変に対する保護の度合をさらに強化するために、バージョン番号をスクランブルすることができる。

【0037】次に、本発明の好ましい実施例によるコンピュータ・システム101上でのソフトウェア・モジュールの動作について述べる。この動作には4つの部分がある。第1の動作は、複数の資格検証トリガを実行可能な目的コード形式のソフトウェア・モジュール中に配置する。第2の動作は、ソフトウェア・モジュールへのアクセスを許可する暗号化された資格付与キー111を生成する。第3の動作は、コンピュータ・システム101が資格付与キー111を受け取り、デコードして、記憶し、製品ロック・テーブル460を設定する。第4の動作は、ソフトウェア・モジュールをシステム101上で

実行して、資格検証トリガに出会ったときには、システム101に資格を検証させる。始めの2つの動作は、ソフトウェア配布者の管理下で実施される。後の2つの動作は、顧客のシステム101上で実施される。

【0038】ソフトウェア配布者は、ソフトウェア・モジュールをコンパイルするとき、資格検証トリガを目的コード中に配置しなければならない。開発用コンピュータ・システム125で起こる典型的なこの過程を図7に示す。原始コードは、資格検証トリガを含めず、通常通りプログラマによって生成される。ステップ701で、原始コードがコンパイラ126に入力され、ステップ702で、プログラム・テンプレートを作成する。プログラム・テンプレートは、仮想計算機レベル404（すなわち機械インターフェース405より上のレベル）の機械語命令を含んでいる。ステップ704で、プログラム・テンプレートは、その製品番号及びバージョン番号を識別すると共に、トランスレータ127への入力として働く。トランスレータ127は、自動的に、かなりの数の資格検証トリガを生成し、これを目的コード中のランダムな位置に挿入し、資格検証トリガの挿入後に参照を解決する。ステップ705で出力された結果得られる実行可能な目的コード形式のソフトウェア・モジュールには、資格検証トリガが含まれている。この実行可能な形式のソフトウェア・モジュールは、実行可能コード・レベル403の目的コード命令を含んでいる。

【0039】暗号化された資格付与キーを生成する過程を図8に示す。ステップ801で、販売担当員によって生成されたあるバージョン・レベルの1個または複数のプログラム製品に対するライセンスの注文が、マーケティング用コンピュータ・システム124に入力される。理論上、顧客が多数のバージョン・レベルの製品を注文することは可能であるが、一般的にはそうする理由はほとんどない。ただし、各資格付与キーは指定されたバージョン・レベルでしか動作しないので、顧客が異なるバージョン・レベルを注文すると、別々の資格付与キーを生成しなければならない。顧客の注文を受け取ると、ステップ802で、システム124で実行中のキー生成/暗号化プログラム122が、顧客に関する情報、具体的には機械の通し番号及びプロセッサ形式の情報を含むデータベース123にアクセスする。この情報を使って、暗号化されていない資格付与キー200の料金グループ・フィールド201及び機械通し番号フィールド204が生成される。ステップ803で、顧客の注文及び可能な製品番号提供のデータベースへの参照によって残りのフィールドが生成されて、完全な非暗号形式の資格付与キーが構築される。次いで、ステップ804で、キー生成/暗号化プログラム122が、当技術分野で既知のいくつかの暗号化技法のうちのどれかを使って、資格付与キーを暗号化する。次いで、ステップ805で、その結果得られた暗号化された資格付与キー111が、顧客に

伝送される。図1ではキー111は複数の2進ビットとして示されているが、資格付与キーを鍵盤から入力する仕事を簡単にするため、16進数字や英数字による等価物など2進ビットをグループ化した何らかの形式で顧客に提示してもよい。

【0040】コンピュータ・システム101上で資格付与キー111を受け取り、製品ロック・テーブル460を維持する過程を図9a及び図9bに示す。ステップ901で、顧客は、操作卓109を介して、資格付与キー111をコンピュータ・システム101に入力する。これが初期導入である場合には、インストール入力ルーチン440がオペレータと対話して入力を受け取る。そうでない場合は、一般入力ルーチン441が入力を受け取る。資格付与キーは、デコード過程を処理するロック解除ルーチン430に渡される。ステップ902で、ロック解除ルーチン430が、機械キー取得機能420に、機械通し番号を検索させ、機械キーを生成させる。次いで、ステップ903で、ロック解除ルーチン430が、機械キーを用いて資格付与キー111をデコードする。次いで、ステップ904で、以下に述べるように、ロック解除ルーチン430がコード化された製品キー・テーブル450を再構築する。デコードされた資格付与キーは、図2に示した形をとる。これは、各製品番号ごとに、その製品が新しい資格付与キーの下でロックを解除されているかどうかを示す、80ビットの製品資格付与フラグ・アレイ205を含んでいる。新しい資格付与キーは、それがロックを解除する全ての製品に対する置換されたキーと見なされる。ロック解除ルーチン430は、デコードされた資格付与キー中の各製品資格付与フラグ205を走査する(ステップ904)。ステップ905で、製品資格付与フラグが“1”(資格付与ありを指す)にセットされている場合、ステップ908で、製品キー・テーブル450中の対応するエントリが、新しい資格付与キー、バージョン番号及び料金グループ値で置換される。資格付与ビット・フィールド506が“1”にセットされ、日/時スタンプ・ビット・フィールド503が、資格付与キーがまだ使用されていないことを示すゼロの値にセットされる。新しいキーの製品資格付与フラグが“0”の場合には、バージョン番号と料金グループ番号が製品キー・テーブル450に記憶されているものと同じでない限り、新しい資格付与キーは効果がない。バージョン番号及び料金グループ番号が同じ場合(ステップ906)、資格付与キーは製品をロックする効果がある。したがって、ロック解除ルーチン430は、ステップ907で、製品ロック・テーブル460中のバージョン番号を“0”にセットするために、ロック設定機能421を呼び出し、ステップ908で、製品キー・テーブル450中の対応するエントリを新しい資格付与キーの値で置換する。製品キー・テーブル450が再構築されると、ステップ909で、その内容が記憶

装置にセーブされる。

【0041】前に資格を付与された製品が資格を失わない限り、製品キー・テーブル450の再構築は製品ロック・テーブル460に直接影響を与えない。製品は、要求に応じて、ロックを解除される。前に資格を付与されていないソフトウェア製品を最初に行うとき、資格検証トリガに出会うと、システムによって例外が生成される。次いで、ステップ920で、例外処理ルーチン432が、製品のロック解除を試みるため、ロック解除ルーチン430を呼び出す。次に、ステップ921で、ロック解除ルーチン430が、コード化された製品キー・テーブル450中の適当なエントリから暗号化された資格付与キーを取り出し、ステップ922で機械キーを得て、ステップ923で資格付与キーを解読する。資格が付与されている(ステップ924)場合は、ステップ926で、ソフトウェア製品の製品番号に対応する、製品ロック・テーブル460のエントリ601中でバージョン番号を設定するために、ロック設定機能421が呼び出される。同時に、ステップ927で、ロック解除ルーチン430が、第1回使用の日時を日/時スタンプ・フィールド503に記録する。次いで、ステップ928で、資格検証トリガが再試行され、プログラムの実行が継続される。ステップ924で資格付与が示されない場合は、ステップ925で、プログラムの実行が打ち切られる。

【0042】製品ロック・テーブル460は、RAMに記憶されており、システムの再初期設定後は残らない。再初期設定(「IPL」)中に、製品ロック・テーブル430を再構築するため、IPLロック再検証ルーチン431が呼び出される。このルーチンは、上記のように、資格を検証し、製品ロック・テーブル460中の対応するエントリを再構築するために機械キーを得て、コード化された製品キー・テーブル450中の各資格付与キー・エントリを系統的にデコードする。

【0043】好ましい実施例による、ソフトウェア・モジュールを実行する過程を図10に示す。システム101によるソフトウェア・モジュールの実行は、モジュールの目的コード命令が終了するまで(ステップ1003)、これを取り出して(ステップ1001)実行する(ステップ1002)ことによってなされる。命令が資格検証トリガ301である(ステップ1004)場合は、ロック検査機能422が呼び出される。ステップ1005で、ロック検査機能422が、資格検証トリガに含まれる製品番号に対応する、製品ロック・テーブル・エントリ601にアクセスする。製品ロック・テーブル460中のバージョン番号が資格検証トリガ301に含まれるバージョン番号303に等しいかまたはそれより大きい場合には、ソフトウェアは実行する資格を付与される(ステップ1006)。この場合、ロック検査機能422はそれ以上の処置を行わず、システムはソフトウ

ウェア・モジュール内の次の目的コード命令の実行に進む。ソフトウェアが資格を付与されていない場合は、ロック検査機能が例外条件を生成して、制御を例外処理ルーチン432に渡し、例外処理ルーチン432がプログラムの実行を終了させる（ステップ1007）。システムは、ソフトウェアが資格を付与されていることを示す資格検査の結果をセーブしない。したがって、ソフトウェア・モジュール中で資格検証トリガに再び出会うと、上記のように、システムは再度資格を検証する。

【0044】別の実施例では、資格検証トリガを、従来のコンパイル経路を介して目的コード中に注入できるように、追加の定義をすることが可能である。これは、顧客及びコンパイラ作成者が利用できる機械インターフェースが、目的コード形式のモジュールがシステムによって実行されるのと同じ実行可能な命令のビットとなっているものとなるはずである。目的コードのフォーマットが顧客に知られている従来のコンパイル経路をサポートするため、資格検証トリガを無効にするかまたは変更するような目的コードの「パッチング」に対する障壁を追加することが適切になる場合がある。このような追加の障壁の一つは、同時に他の何らかの機能を果たすように、資格検証トリガを定義することである。この場合、資格検証トリガによって実施される代替機能が、他の単純命令で実施できないことが肝要である。この代替機能は、コンパイルされたどんなソフトウェア・モジュールもかなり確実にその機能を果たすいくつかの命令を含むように、選定しなければならない。これらの判定規準に合致している場合、コンパイラは、自動的に、その通常のコンパイル手順の一部として、その代替機能を

（それと同時に資格検証トリガも）果たす目的コードを生成することができる。この定義は、資格検証トリガを無効にするような目的コードの「パッチング」に対する重要な障壁をもたらすはずである。他の実施例は、資格検証トリガを、目的コード中で、それが識別する製品番号に対して単純な関係をもつアドレス可能位置に、配置しなければならないと定義することである。コンパイラが、通常のコンパイル過程の一部としてこれらの命令を適切なコード位置に注入するのは比較的簡単なはずであり、命令実施態様でこの追加的検証を行うのは簡単にはずである。この定義は、資格検証トリガ中で供給される製品番号の識別を変更する、目的コードのパッチングに対する重要な障壁となるはずである。

【0045】好ましい実施例は、80個の独立な製品番号に対応している。本発明によってサポートされる製品番号の実際数は可変であることを理解されたい。好ましい実施例では、配布者から配布される、別々にコンパイル可能なソフトウェア・モジュールの数が80を大幅に越えることも予想されている。製品番号の数は、配布者のマーケティング組織によって提供される別々に価格設定されたソフトウェア・パッケージの数に対応する。

各ソフトウェア・モジュールは一つの製品番号しかもたないが、この製品番号を共有するソフトウェア・モジュールは多数存在し得る。たとえば、配布者は、画面編集、スペル・チェック、文書フォーマット化などを扱う別々のソフトウェア・モジュールを含む、ワード・プロセッシング・パッケージを提供する。こうしたソフトウェア・モジュールが常にワード・プロセッシング・パッケージの一部としてライセンスを与えられる場合には、それらのモジュールは共通の製品番号をもつことになる。別の実施例では、各ソフトウェア・モジュールごとに、別々の番号をもつことも可能である。

【0046】好ましい実施例では、ソフトウェアは一括払い料金でライセンスが与えられるが、グレードアップを維持するための追加料金が請求されることが可能である。別の実施例では、ある期間の間ソフトウェアのライセンスを許諾することができる。このような別の実施例においては、資格付与キーが、ソフトウェアのライセンスが与えられる期間の長さを示す追加のフィールドを含むことになる。コード化された製品キー・テーブル450中の製品番号に関連する日/時スタンプをライセンス許諾の期間の長さと比較して、ライセンスが満了したかどうかを判定するために、IPLロック再検証ルーチン431が、定期的に呼び出される。このとき、ライセンスが満了した場合には、製品ロック・テーブル460中の対応するエントリにロック解除ビット（図示せず）を設けておきこれを“0”にセットすることによって、新たな資格付与が得られるまで、ソフトウェア・モジュールのそれ以上の実行を防止するようにすることもできる。

【発明の効果】本発明によって、コンピュータ・システムにおけるソフトウェアの使用を管理する、改善された方法及び装置が提供される。

【図面の簡単な説明】

【図1】本発明の好ましい実施例によるソフトウェア保護機構の主要要素を示す図である。

【図2】本発明の好ましい実施例による資格付与キーの暗号化されていない内容を示す図である。

【図3】好ましい実施例による典型的な実行可能ソフトウェア・モジュールの内容を示す図である。

【図4】好ましい実施例によるソフトウェア保護機構をサポートするために顧客のコンピュータ・システム上で必要なハードウェア及びソフトウェアの構造を示す図である。

【図5】好ましい実施例によるコード化された製品キー・テーブルのフォーマットを示す図である。

【図6】好ましい実施例による製品ロック・テーブルのフォーマットを示す図である。

【図7】好ましい実施例による、資格検証トリガをソフトウェア・モジュールに配置するのに必要なステップのブロック図である。

【図8】好ましい実施例により、暗号化された資格付与キーを生成するのに必要なステップのブロック図である。

【図9】好ましい実施例による、顧客のコンピュータ・システム上で資格付与キーをデコードし、資格付与状況の記録を維持するのに必要なステップのブロック図である。

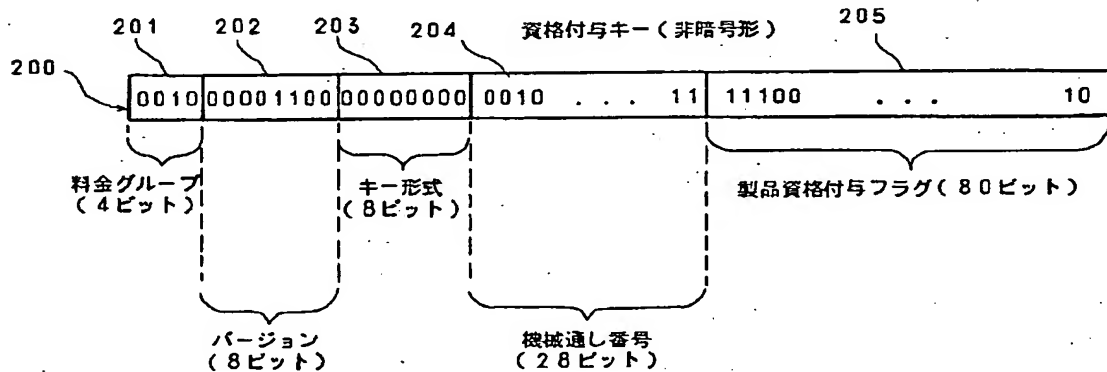
【図10】好ましい実施例による、ソフトウェア・モジュールの実行中に資格を検証するのに必要なステップのブロック図である。

【符号の説明】

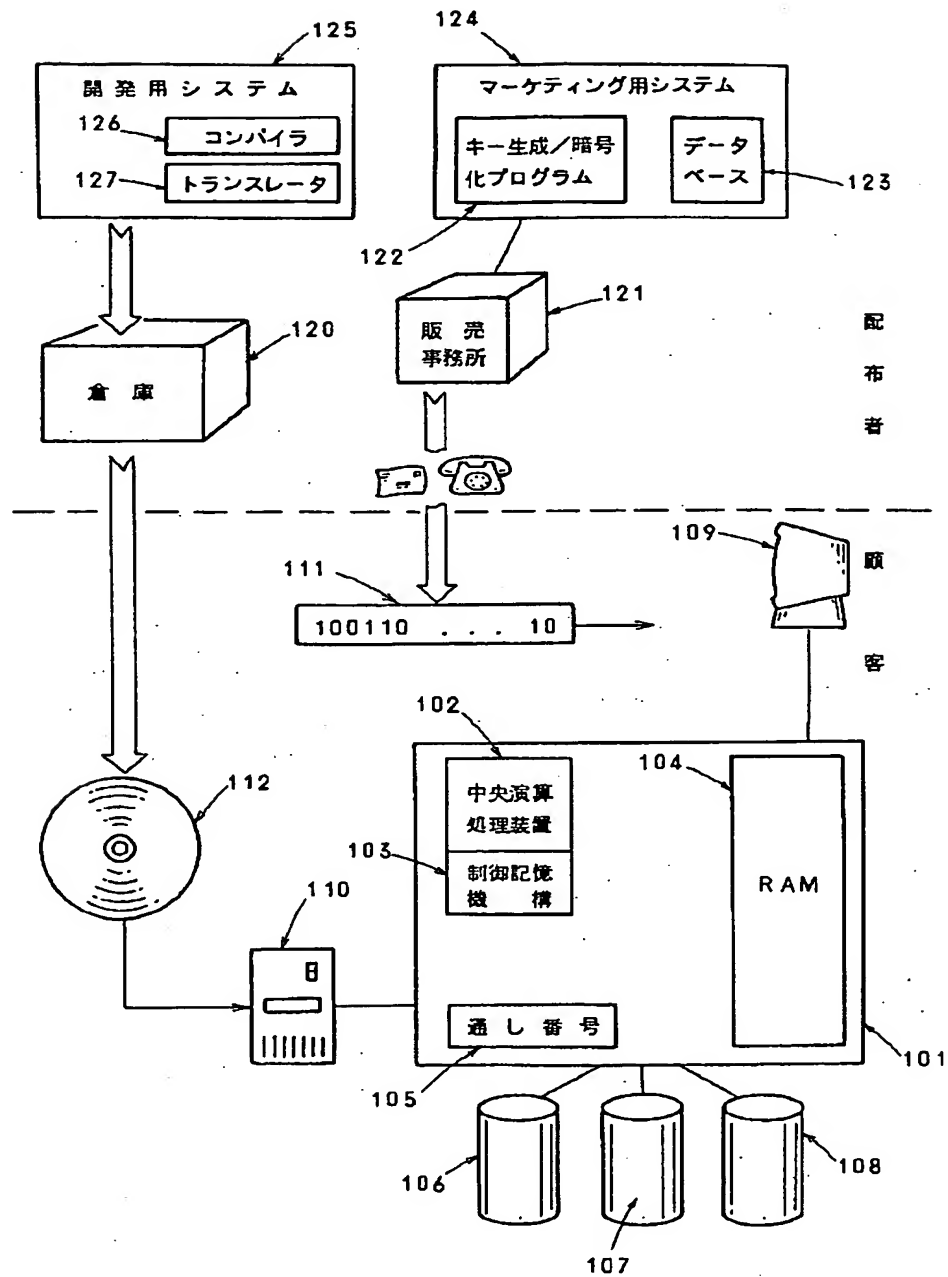
- 101 顧客側コンピュータ・システム
- 102 中央演算処理装置 (CPU)
- 103 制御記憶機構

- * 104 ランダム・アクセス・メモリ (RAM)
- 106 記憶装置
- 107 記憶装置
- 108 記憶装置
- 109 操作卓
- 110 媒体読取装置
- 112 ソフトウェア記録媒体
- 120 倉庫
- 121 販売事務所
- 10 123 データ・ベース
- 124 マーケティング用コンピュータ・システム
- 125 開発用コンピュータ・システム
- 126 コンパイラ
- * 127 トランスレータ

【図2】

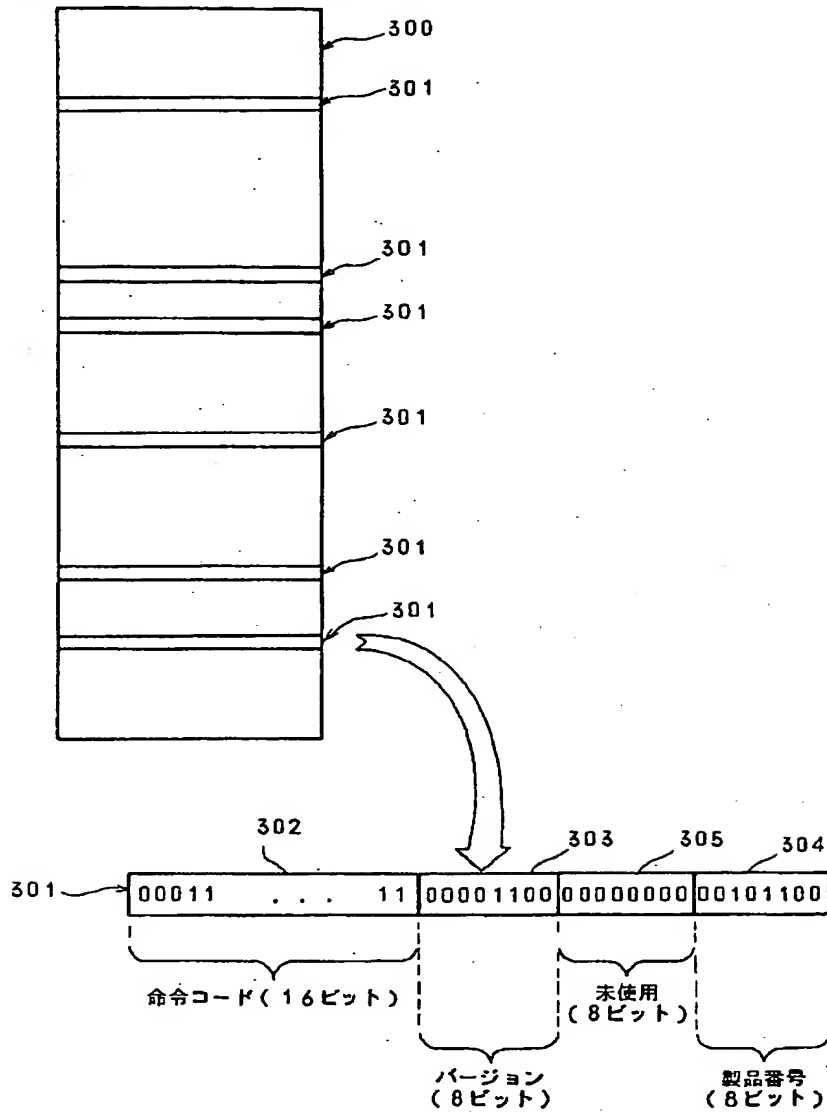


【図1】

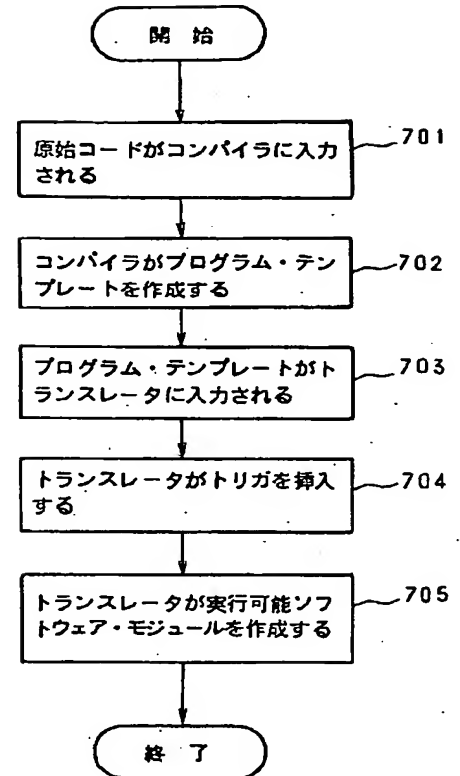


【図3】

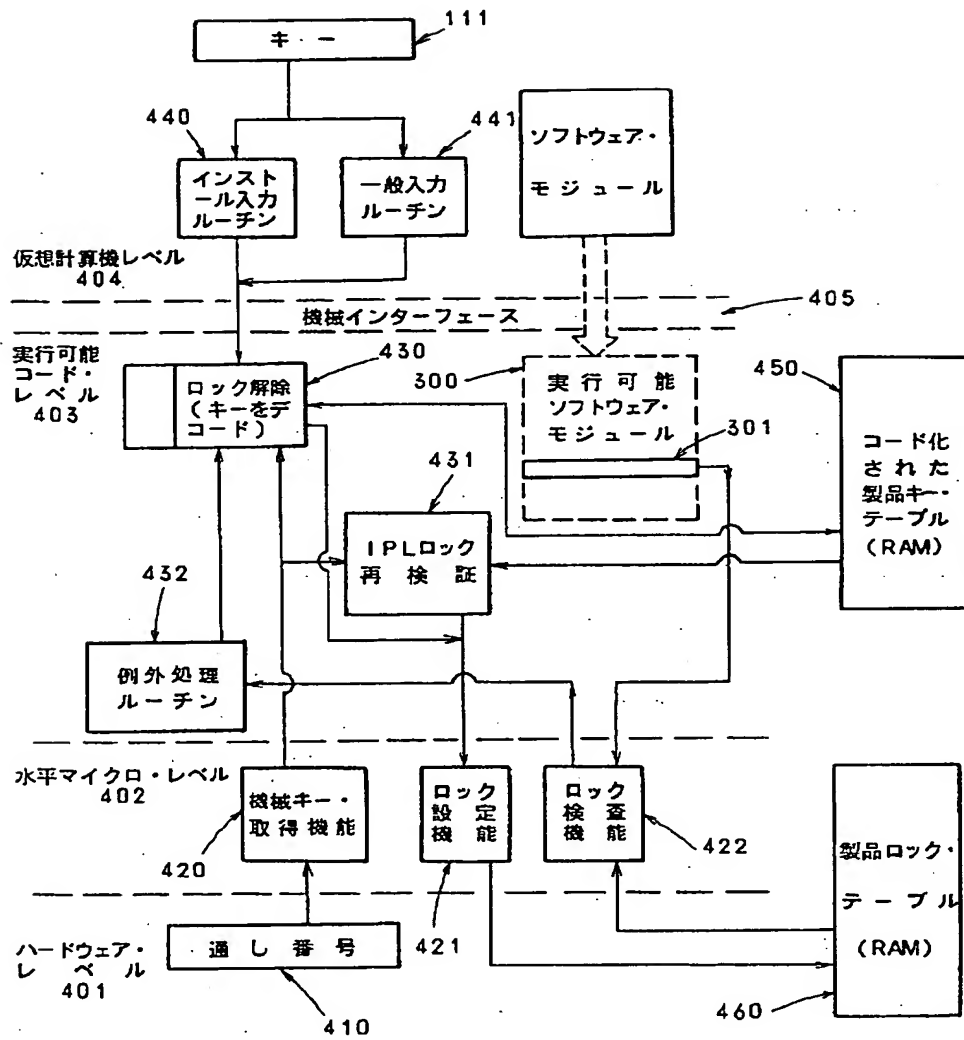
実行可能ソフトウェア・モジュール



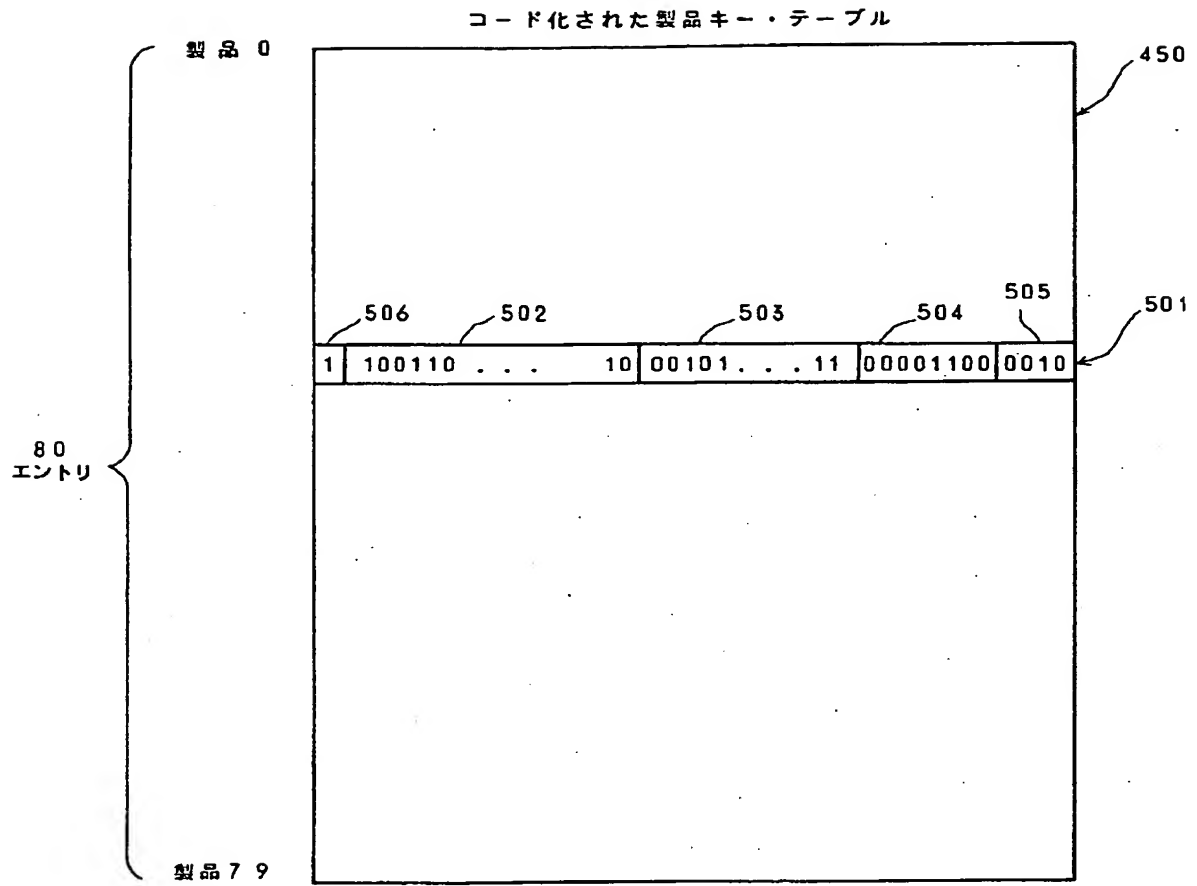
【図7】



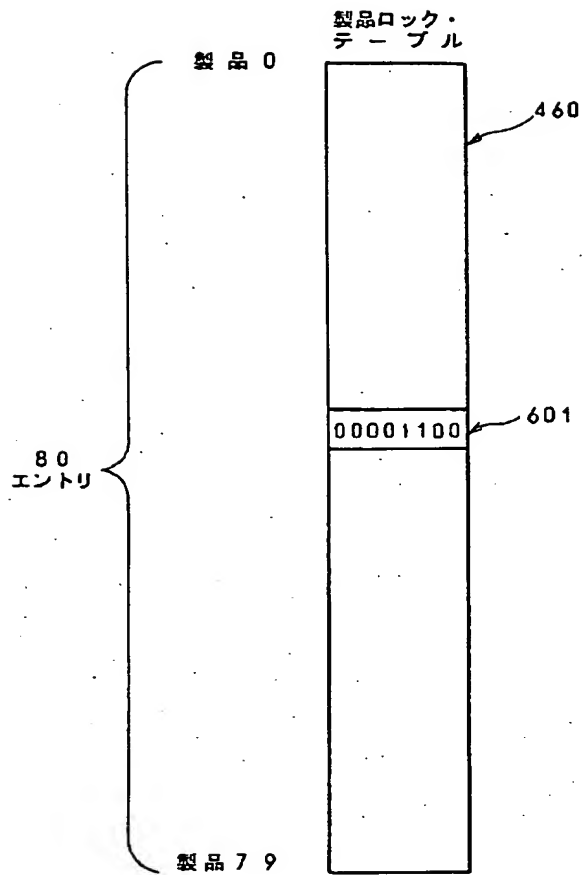
【図4】



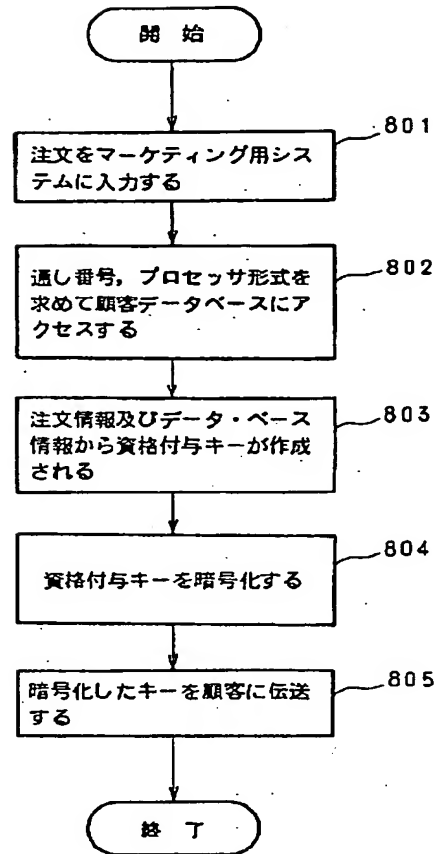
【図5】



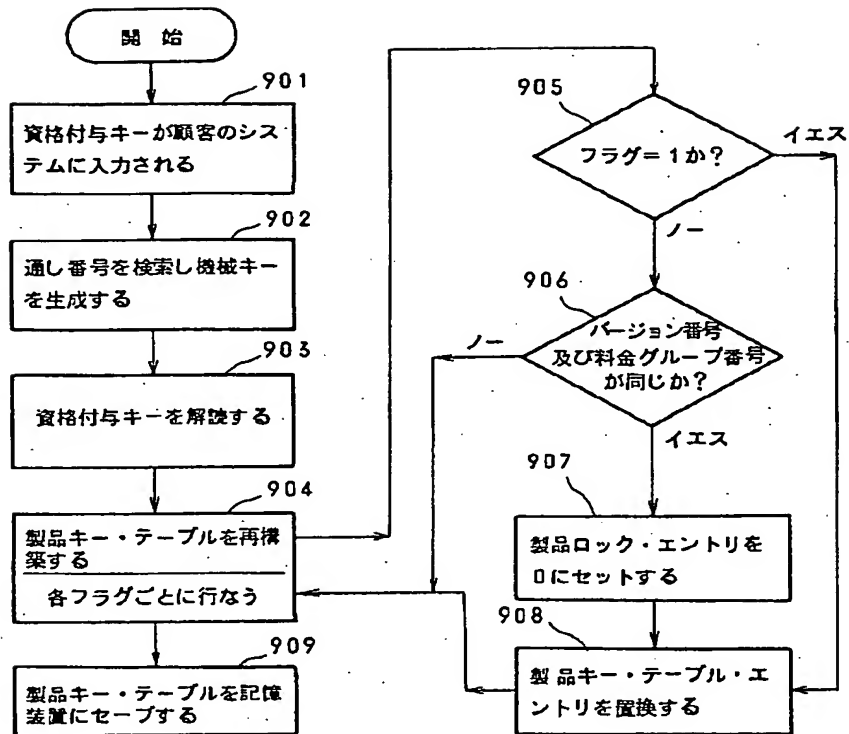
【図6】



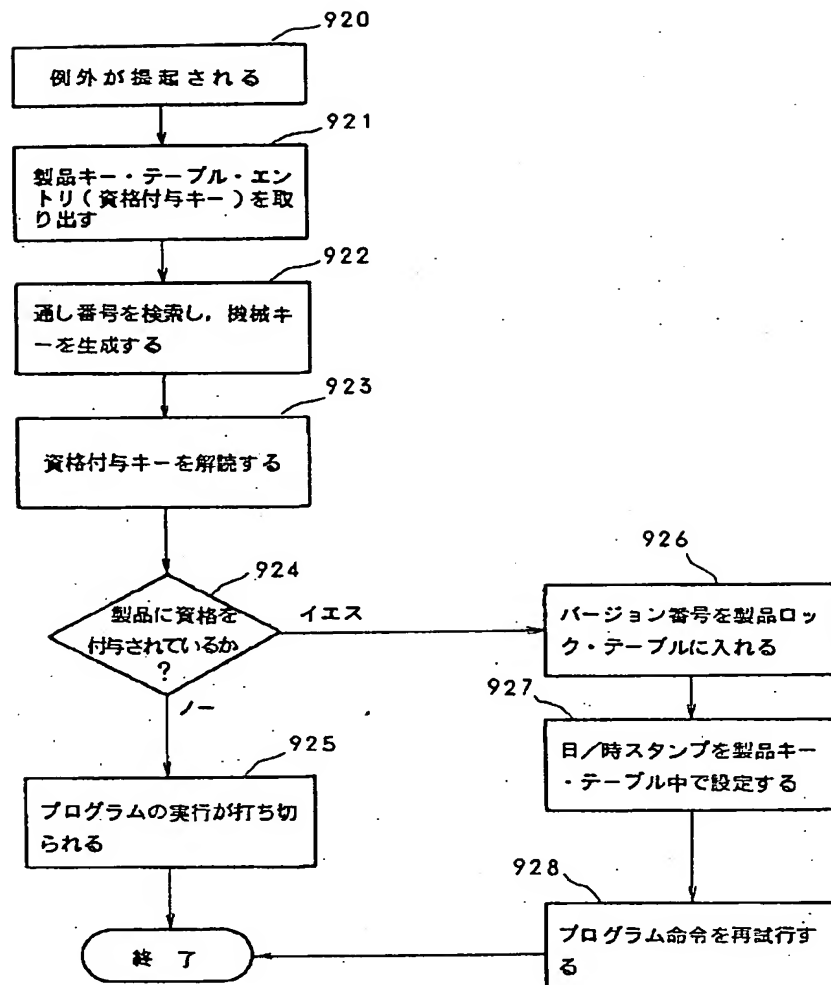
【図8】



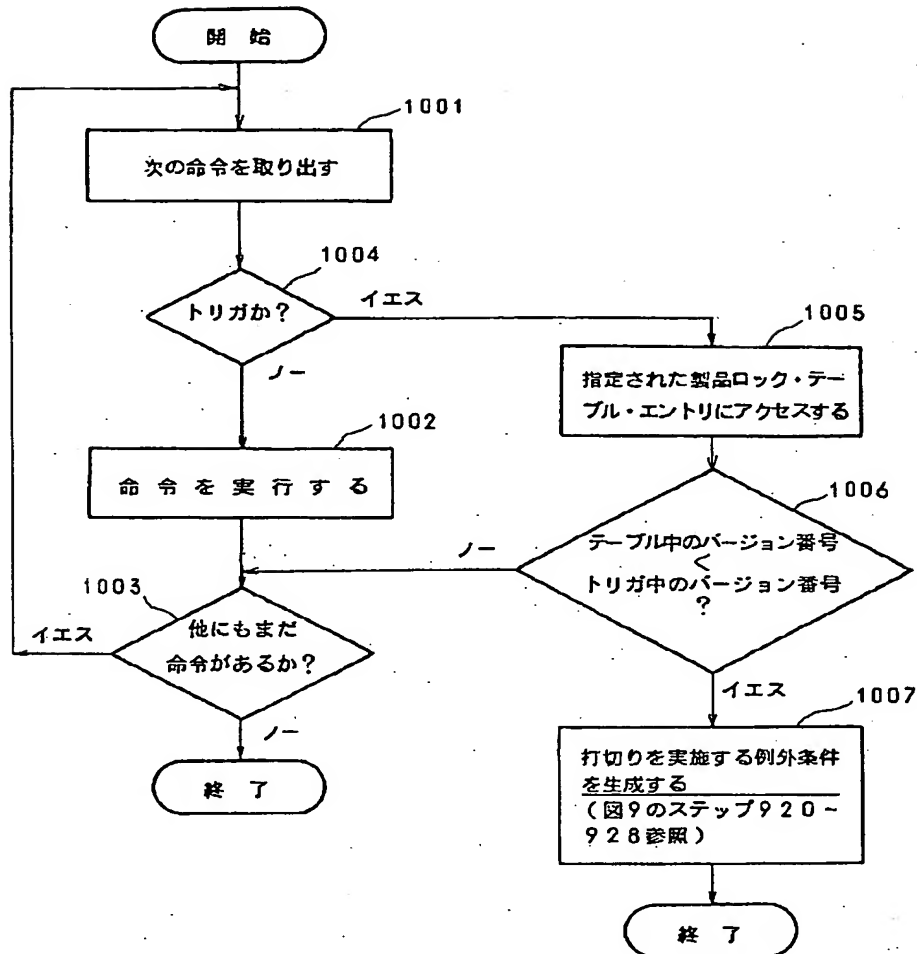
【図9a】



【図9b】



【図10】



フロントページの続き

(72)発明者 マイケル・ジョーゼフ・コリガン
アメリカ合衆国55906、ミネソタ州ロチェ
スター、20番アベニュー、ノース・イース
ト 1510番地

(72)発明者 フランシス・ジョーゼフ・リアドン・ジュ
ニア
アメリカ合衆国55901、ミネソタ州ロチェ
スター、シャトー・ロード、ノース・ウェ
スト 5685番地

(72)発明者 ジェームズ・ウィリアム・モラン
アメリカ合衆国55934、ミネソタ州エヨタ、
チェスター・ロード、サウス・イースト、
3221番地